Интеллектуальные системы и искусственный интеллект (1)

Искусственный интеллект (ИИ) — свойство интеллектуальных систем решать задачи, требующие для своего решения человеческого интеллекта (обучение, рассуждение, восприятие, принятие решений) в условиях неопределенности.

Интеллектуальная система — программно-аппаратный комплекс, реализующий функции интеллектуального поведения (анализ, синтез, обобщение) при решении задач в условиях неполноты или противоречивости исходной информации.

Академические основы и архитектурная эволюция

План лекции: ключевые направления ИИ (2)

- Компьютерное зрение: детекция, сегментация, генерация изображений
- Обработка естественного языка: трансформеры, языковые модели
- Обучение с подкреплением: робототехника, оптимизация решений
- Генеративные модели: GAN, диффузия, автоэнкодеры
- Эволюция архитектур: от RNN до Mamba
- Будущее ИИ: эффективность, интерпретируемость, этика

Шесть фундаментальных направлений современного ИИ

Основные определения (3)

Искусственный интеллект (ИИ) — свойство интеллектуальных систем решать задачи, требующие человеческого интеллекта (обучение, рассуждение, восприятие) в условиях неопределенности.

Интеллектуальная система — программно-аппаратный комплекс, реализующий функции интеллектуального поведения при решении задач в условиях неполноты или противоречивости информации.

Иерархия ИИ

Искусственный интеллект (ИИ)

↓

Интеллектуальные системы

↓

Символьные системы
(дедуктивный подход)
• Экспертные системы
• Логическое программирование

↓

Машинное обучение
(индуктивный подход)
• Нейросети
• Алгоритмы обучения

↓

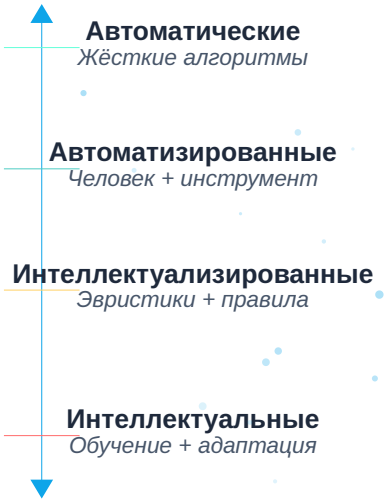
Гибридные системы
(комбинированный подход)
• Нейро-символьные
• Мультимодальные

Каждый уровень — подмножество вышестоящего

Классификация систем по степени интеллектуальности (4)

Тип системы	Основа функционирования	Адаптация	Примеры
Автоматическая	Жёсткие алгоритмы	Отсутствует	Термостат, конвейер
Автоматизированная	Человек + инструменты	Человеком	CAD-системы
Интеллектуализированная	Эвристики + правила	Ограниченная	Системы поддержки решений
Интеллектуальная	Моделирование интеллекта	Автономная	Нейросети, агенты с обучением

Ключевой критерий: способность к автономному обучению и адаптации без вмешательства человека.



Ключевой критерий интеллектуальности:
способность к автономному обучению и адаптации без вмешательства человека

Статические и динамические интеллектуальные системы (5)

Статические системы (интеллектуализированные)
- Однократная обработка входных данных
- Фиксированный алгоритм преобразования
- Отсутствие памяти и обратной связи
- Пример: экспертный вывод, компилятор

Динамические системы (интеллектуальные)
- Непрерывное взаимодействие со средой
- Адаптация на основе обратной связи
- Наличие внутреннего состояния и памяти
- Пример: агенты с обучением, рекуррентные сети

Фундаментальное различие: наличие механизма обратной связи и способность к изменению состояния на основе опыта.

Динамическая система: $\mathbf{s}_{t+1} = f(\mathbf{s}_t, \mathbf{x}_t), \quad \mathbf{y}_t = g(\mathbf{s}_t, \mathbf{x}_t)$

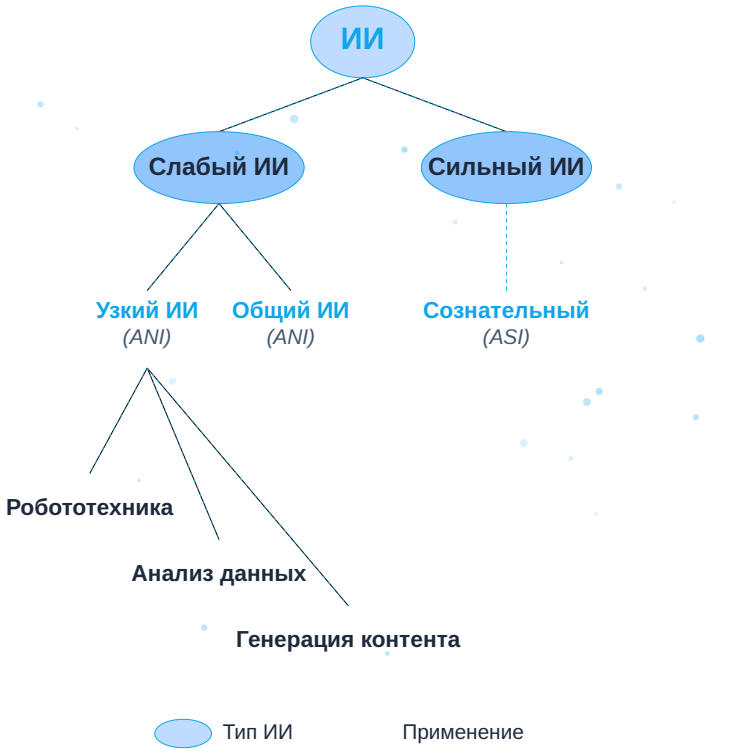
Два уровня «интеллекта»

Фундаментальное различие:
наличие обратной связи и адаптация на опыте

Иерархия искусственного интеллекта (6)

Искусственный интеллект (ИИ)
└── **Интеллектуальные системы**
    ├── **Символьные системы** (дедуктивный подход)
    │   ├── Экспертные системы (базы знаний + логический вывод)
    │   └── Системы логического программирования
    │
    └── **Системы машинного обучения** (индуктивный подход)
        ├── Обучение с учителем (супервизированное)
        │   ├── Классификация
        │   └── Регрессия
        │
        ├── Обучение без учителя (несупервизированное)
        │   ├── Кластеризация
        │   └── Снижение размерности
        │
        └── Обучение с подкреплением
            └── Оптимизация поведения через взаимодействие со средой

Классификация интеллектуальных систем



Дедуктивный подход: экспертные системы (7)

Экспертная система — программа, использующая базу знаний и механизм логического вывода для решения задач в узкой предметной области на уровне эксперта.

Архитектура:

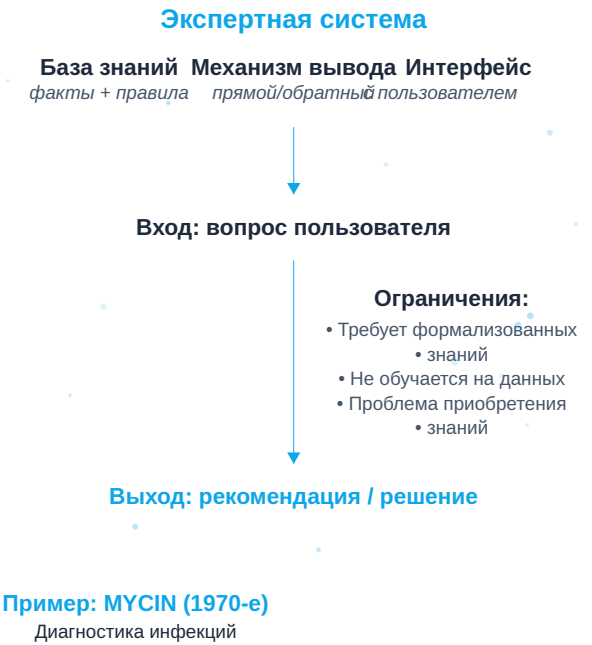
- База знаний: факты и правила («если-то»)
- Механизм вывода: прямой/обратный вывод
- Интерфейс с пользователем

Ограничения:

- Требует явного формализованного знания
- Не способна к обучению на данных
- Проблема приобретения знаний

Пример: MYCIN (диагностика бактериальных инфекций, 1970-е).

Архитектура экспертной системы



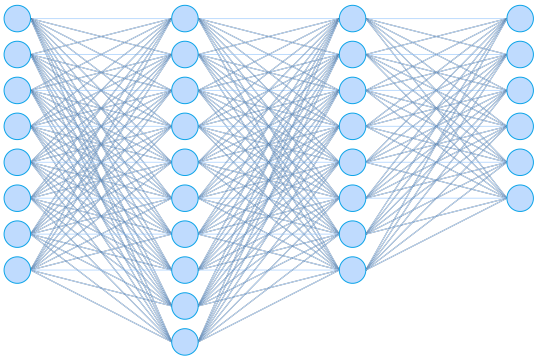
Пример: MYCIN (1970-е)

Диагностика инфекций

Индуктивный подход: машинное обучение (8)

Машинное обучение (МО) — подраздел ИИ, изучающий алгоритмы, способные обучаться на данных без явного программирования правил.
Фундаментальная идея: при наличии достаточного объёма данных алгоритм МО автоматически извлекает закономерности и строит функцию отображения вход → выход.
Преимущество перед дедуктивными системами: способность к обобщению на новые данные без изменения архитектуры системы.
Пример: нейронные сети как универсальные аппроксиматоры (теорема Цыбенко).

Нейронная сеть



Теоремы универсальной аппроксимации (9)

Теорема Колмогорова-Арнольда (1957):
Любая многомерная непрерывная функция представима как суперпозиция функций одной переменной.

$$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

Теорема Цыбенко (1989):
Сеть с одним скрытым слоем и сигмоидальной активацией аппроксимирует любую непрерывную функцию на компакте.

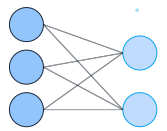
$$\forall f \in C(I_n), \forall \epsilon > 0, \exists F(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \sigma(\mathbf{w}_j^T \mathbf{x} + b_j)$$

$$\sup_{\mathbf{x} \in I_n} |F(\mathbf{x}) - f(\mathbf{x})| < \epsilon$$

Практическое ограничение: Теоремы гарантируют существование, но не эффективность — число нейронов может быть экспоненциально большим.

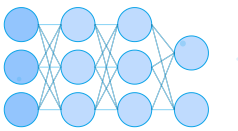
Сравнение архитектур нейронных сетей

Однослойная сеть
(Shallow Network)



Сложность: O(2ⁿ)
экспоненциальный рост

Многослойная сеть
(Deep Network)



Сложность: O(n·k)
линейный рост

Глубокие сети эффективнее для сложных функций
иерархическое извлечение признаков с меньшими затратами

Преимущество глубины архитектур (10)

Экспоненциальная сложность плоских сетей:
Для представления иерархических функций однослойной сети требуется экспоненциальное число нейронов.

$$N_{shallow} \sim O(2^n)$$

Линейная сложность глубоких сетей:
Глубокая архитектура решает ту же задачу с полиномиальным числом параметров.

$$N_{deep} \sim O(n \cdot k)$$

где n — размерность входа, k — глубина сети.

Примеры глубоких архитектур:

- **ResNet-152:** 152 слоя для классификации изображений (ImageNet)
- **BERT:** 12-24 слоя трансформера для обработки языка
- **GPT-3:** 96 слоёв, 175 млрд параметров

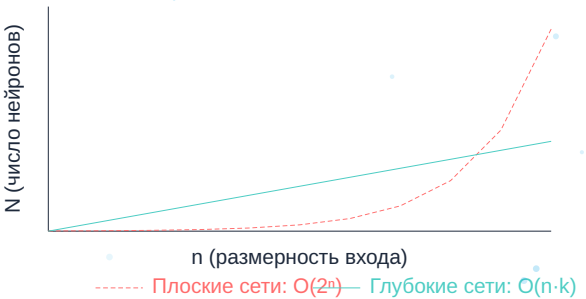
Вывод: Глубина позволяет выделять иерархические признаки с меньшей вычислительной стоимостью.

Глубина решает: экспонента vs полином

Плоские сети
Экспоненциальная сложность:
$N_{hhalxhnə} \sim O(2^n)$

Глубокие сети
Линейная сложность:
$N_{eeem} \sim O(n \cdot k)$

n — размерность входа, k — глубина



Примеры глубоких архитектур:

ResNet-152	BERT	GPT-3
152 слоя	12-24 слоя	96 слоёв
Классификация ImageNet	*Обработка языка*	*175 млрд параметров*

Глубина позволяет выделять иерархические признаки с меньшей вычислительной стоимостью

Теорема Эльдана-Шамира о глубине (11)

Теорема (Eldan & Shamir, 2016):
Существуют функции, которые глубокая сеть с 3 слоями может представить точно, а сеть с 2 слоями требует экспоненциального числа нейронов.

Математическая формулировка:
Существует распределение $p(\mathbf{x})$ и функция $f : \mathbb{R}^d \to \mathbb{R}$ такая что:

$$\exists \text{ 3-layer network} : N_3 = O(d)$$

$$\forall \text{ 2-layer network} : N_2 = \Omega(e^d)$$

Интуиция:
Глубокие сети эффективно представляют композицию функций $f = f_L \circ f_{L-1} \circ \cdots \circ f_1$

Практическое значение:
• Объясняет эффективность глубоких архитектур в реальных задачах
• Глубина ≠ просто больше параметров, а качественное изменение выразительности
• Основа для проектирования архитектур ResNet, DenseNet, Transformer

Теорема Эльдана-Шамира о глубине (201

Существуют функции, которые глубокая сеть с 3 слоями может представить точно, а сеть с 2 слоями требует экспоненциального ч нейронов.

Математическая формулировка:
Существует распределение p(x) и функция f : R^d → R такая

∃ 3-layer network : N_3 = O(d)

∀ 2-layer network : N_2 = Ω(e^d)

Визуальное сравнение:

2 слоя: N_2 = Ω(e^d) 3 слоя: N_3 = O(d)

Интуиция:
Глубокие сети эффективно представляют композицию функ

f = f_L ∘ f_{L-1} ∘ ⋯ ∘ f_1

f1 → f2 → f3 → f4 → f5

Практическое значение:
• Объясняет эффективность глубоких архитектур в реальнь
• Глубина ≠ просто больше параметров, а качественное изм
• Основа для проектирования архитектур ResNet, DenseNet

Три парадигмы машинного обучения (12)

Обучение с учителем

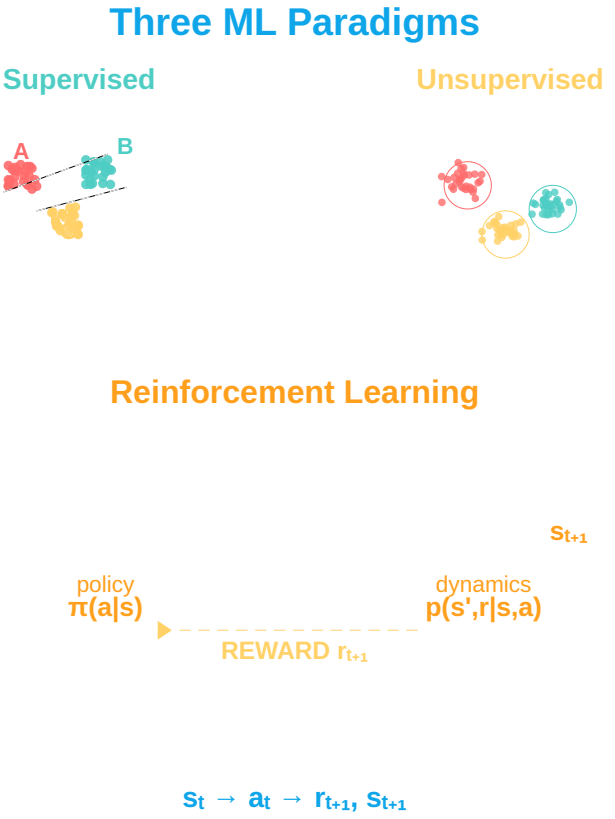
- Обучающая выборка: пары $(\mathbf{x}_i, y_i)$
- Цель: найти функцию $f : \mathcal{X} \to \mathcal{Y}$, минимизирующую ошибку
- Примеры: классификация изображений, предсказание цен

Обучение без учителя

- Обучающая выборка: только $\{\mathbf{x}_i\}_{i=1}^{N}$
- Цель: выявление скрытой структуры данных (кластеры, многообразия)
- Примеры: кластеризация клиентов, снижение размерности

Обучение с подкреплением

- Взаимодействие агента со средой через действия a_t
- Обратная связь: скалярная награда r_{t+1}
- Цель: максимизация дисконтированной совокупной награды



Обучение с учителем: регрессия и классификация (13)

Регрессия

- Выходное пространство: $\mathcal{Y} \subseteq \mathbb{R}^d$ (непрерывное)
- Функция потерь: среднеквадратичная ошибка (MSE)
- Пример: предсказание цены недвижимости

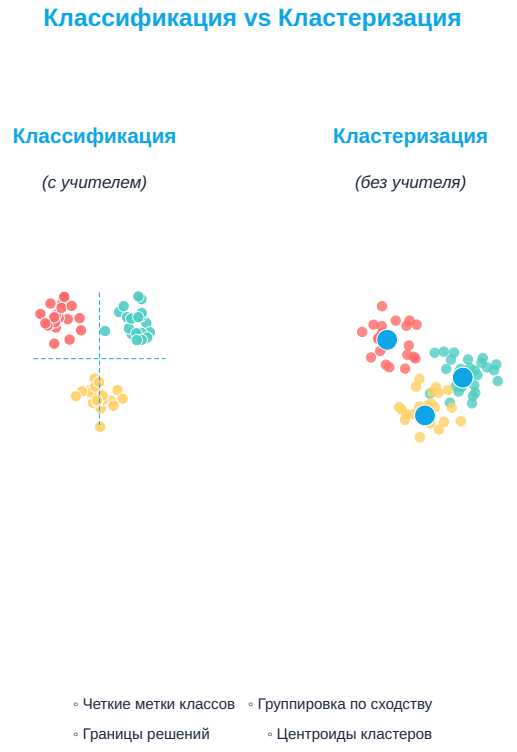
Классификация

- Выходное пространство: $\mathcal{Y} = \{1, \ldots, C\}$ (дискретное)
- Функция потерь: кросс-энтропия
- Пример: распознавание цифр (MNIST)

Единая формулировка:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i)) + \lambda \Omega(f)$$

где $\Omega(f)$ — регуляризатор, предотвращающий переобучение.



Градиентный спуск (14)

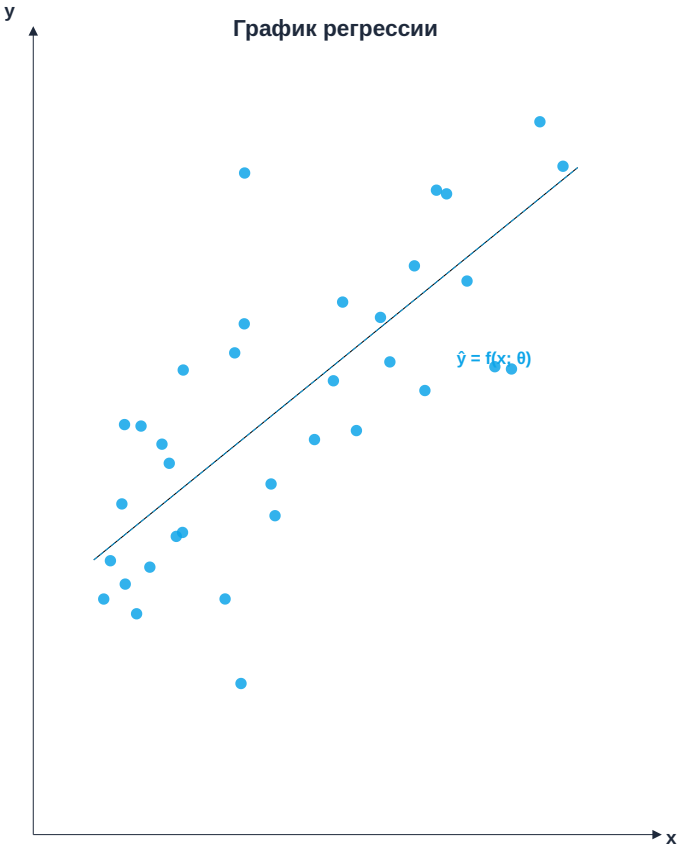
Градиентный спуск — итеративный алгоритм минимизации функции потерь $J(\theta)$:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta J(\theta_t)$$

Где:
- θ — параметры модели
- $\eta > 0$ — скорость обучения
- $\nabla_\theta J$ — градиент функции потерь

Интуиция: движение в направлении наискорейшего спуска по поверхности ошибки.

Варианты: стохастический (SGD), с моментом (Momentum), Adam.



Обучение без учителя: кластеризация (15)

Кластеризация — группировка объектов так, чтобы объекты внутри кластера были более похожи друг на друга, чем на объекты из других кластеров.

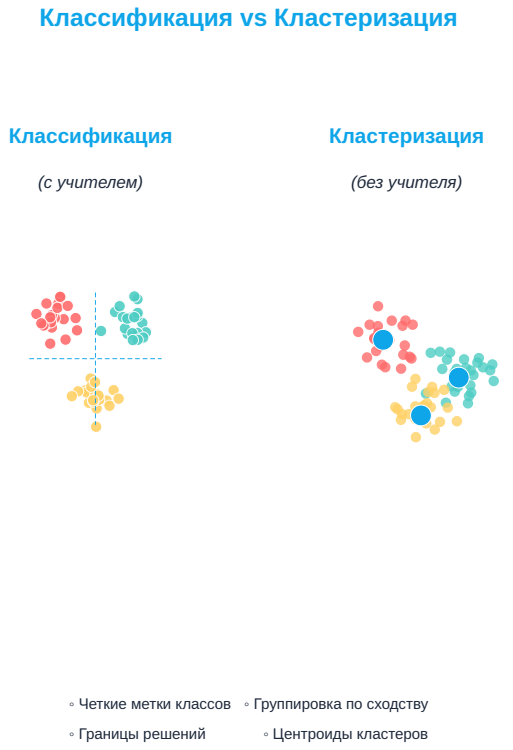
Метрики качества:

- Силуэт: $s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$
- Внутрикластерная дисперсия

Алгоритмы:

- K-means: минимизация внутрикластерной дисперсии
- Иерархическая кластеризация: построение дендрограммы
- DBSCAN: обнаружение кластеров произвольной формы

Применения: сегментация рынка, обнаружение аномалий.



Нейронные сети: основа индуктивного подхода (16)

Нейронная сеть — вычислительная модель, реализующая индуктивный вывод на основе обучающих примеров.

Математическая модель нейрона:

$$a = \sigma\left(\mathbf{w}^T\mathbf{x} + b\right)$$

где $\sigma(\cdot)$ — нелинейная функция активации (ReLU, sigmoid, tanh).

Архитектурные компоненты:

- Входной слой: кодирование исходных данных
- Скрытые слои: иерархическое извлечение признаков
- Выходной слой: формирование результата

Теорема универсальной аппроксимации: полносвязная сеть с одним скрытым слоем может аппроксимировать любую непрерывную функцию с произвольной точностью.

Автоматическое кодирование (автоэнкодеры) (17)

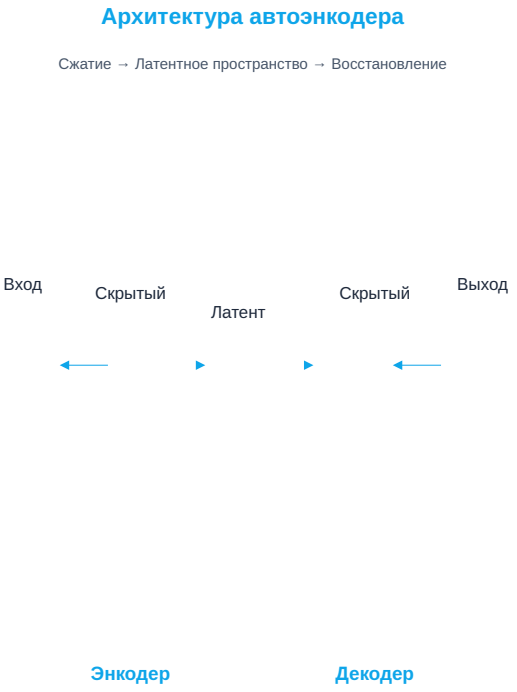
Автоэнкодер — архитектура для нелинейного снижения размерности через «бутылочное горлышко»:

Энкодер: преобразует вход $\mathbf{x} \in \mathbb{R}^D$ в компактное скрытое представление $\mathbf{z} = f_\phi(\mathbf{x}) \in \mathbb{R}^d,\ d \ll D$

Бутылочное горлышко: узкое скрытое пространство, фиксирующее существенные признаки

Декодер: восстанавливает данные из кода $\hat{\mathbf{x}} = g_\theta(\mathbf{z})$

Применения: сжатие данных, денойзинг, генерация.



Сравнение автоэнкодера и вариационного автоэнкодера (18)

Характеристика	Автоэнкодер (AE)	Вариационный автоэнкодер (VAE)
Скрытое пространство	Детерминированное	Вероятностное
Генерация	Ограниченная	Высококачественная
Обучение	Минимизация ошибки реконструкции	Максимизация ELBO
Интерполяция	Нелинейная	Плавная и семантически осмысленная

Ключевое уравнение VAE (ELBO):

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

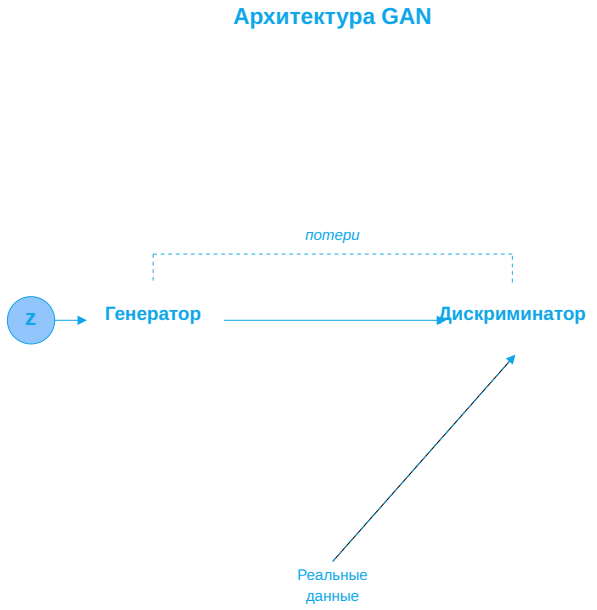
Генеративно-состязательные сети (GAN) (19)

GAN реализует минимаксную игру между двумя сетями:
Генератор G_θ: отображает шум $\mathbf{z} \sim p_z(\mathbf{z})$ в пространство данных $\mathbf{x}' = G_\theta(\mathbf{z})$
Дискриминатор D_ϕ: оценивает вероятность принадлежности образца реальным данным
Целевая функция:

$$\min_\theta \max_\phi V(D_\phi, G_\theta) = \mathbb{E}_{\mathbf{x}\sim p_{data}}[\log D_\phi(\mathbf{x})] +$$

$$+\mathbb{E}_{\mathbf{z}\sim p_z}[\log(1 - D_\phi(G_\theta(\mathbf{z})))]$$

Равновесие: генератор производит неотличимые от реальных данные ($D_\phi(\mathbf{x}) = 0.5$).



Диффузионные модели (20)

Диффузионные модели основаны на постепенном разрушении и восстановлении структуры данных:

Прямой процесс: последовательное добавление шума к данным

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Обратный процесс (обучаемый): постепенное удаление шума

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$$

Преимущества:

- Теоретически обоснованная сходимость
- Отсутствие проблемы коллапса мод (в отличие от GAN)
- Высокое качество генерации

Поток генерации изображений (21)

Процесс генерации изображения в диффузионных моделях:

1. **Старт ($t = T$):** случайный шум $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$
2. **Итерация t:** модель $\epsilon_\theta(\mathbf{x}_t, t)$ предсказывает шум
3. **Денойзинг:** $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
4. **Результат ($t = 0$):** фотореалистичное изображение $\mathbf{x}_0$

Каждая итерация уточняет изображение, постепенно удаляя шум и добавляя детали.

Генерация контента с помощью ИИ

Модели, обучающиеся распределению данных для создания новых, правдоподобных примеров

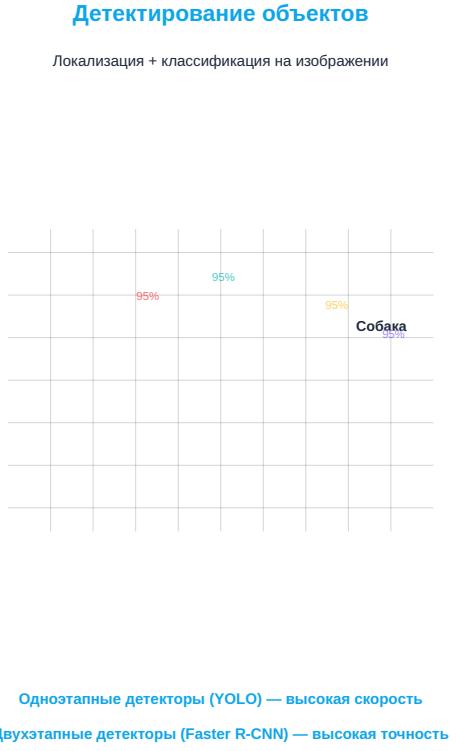
Детектирование объектов (22)

Детектирование объектов решает две задачи в едином конвейере:
1. **Локализация:** определение положения объекта через ограничивающий бокс $\mathbf{b} = (x_{min}, y_{min}, x_{max}, y_{max})$
2. **Классификация:** определение категории объекта внутри бокса $c \in \{1, \ldots, C\}$

Метрика качества: Intersection over Union (IoU)

$$\text{IoU}(\mathbf{b}_{pred}, \mathbf{b}_{gt}) = \frac{|\mathbf{b}_{pred} \cap \mathbf{b}_{gt}|}{|\mathbf{b}_{pred} \cup \mathbf{b}_{gt}|}$$

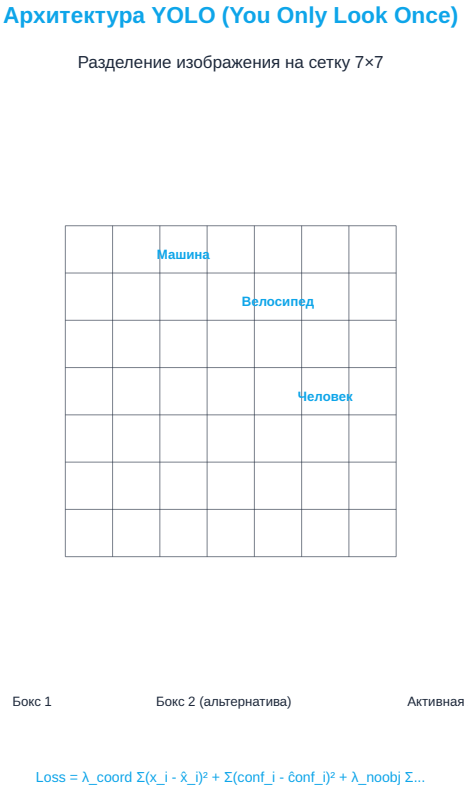
Архитектурные подходы:
- Двухступенчатые (Faster R-CNN)
- Одноступенчатые (YOLO, SSD)
- Трансформерные (DETR)



YOLO: единый проход для детекции (23)

Принцип работы: разделение изображения на регулярную сетку $S \times S$, где каждая ячейка предсказывает:
- B ограничивающих боксов с координатами (x, y, w, h) и уверенностью

Преимущества: скорость (реальное время), глобальный контекст, простота архитектуры.



Сегментация изображений (24)

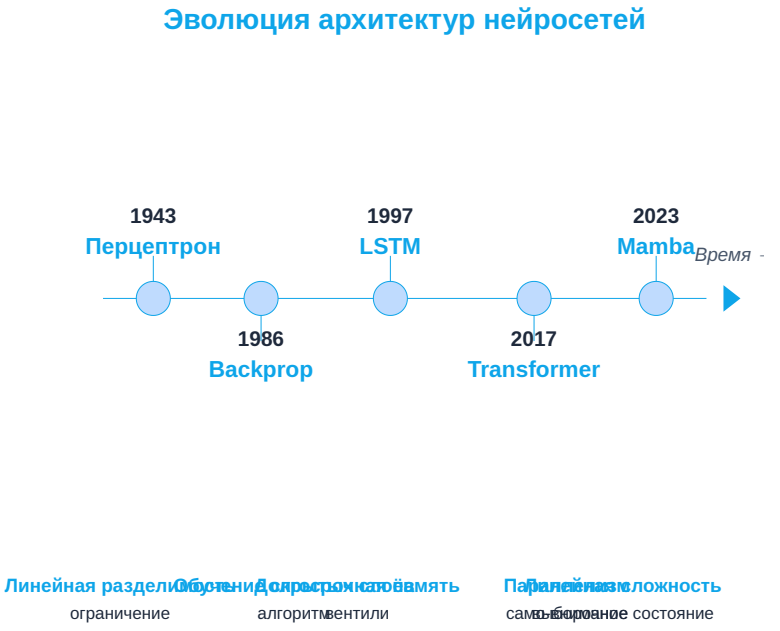
Семантическая сегментация:
- Каждому пикселю присваивается метка класса
- Все объекты одного класса неразличимы
- Архитектура: FCN, U-Net
- Метрика: mean IoU

Сегментация по инстансам:
- Каждый объект получает уникальный идентификатор
- Разделение объектов одного класса на отдельные инстансы
- Подходы: Mask R-CNN, кластеризация по признакам
- Метрика: Average Precision по маскам

Применения: автономные автомобили, медицинская визуализация, робототехника.

Хронология развития архитектур глубокого обучения (25)

Год	Архитектура	Ключевой вклад	Ограничение
1943	Перцептрон	Биологическая модель нейрона	Линейная разделимость
1986	Обратное распространение	Обучение многослойных сетей	Затухающие градиенты
1997	LSTM	Долгосрочная память через вентили	Вычислительная сложность
2012	AlexNet	Глубокие свёрточные сети	Требовательность к данным
2017	Трансформер	Параллельная обработка через внимание	Квадратичная сложность $O(N^2)$
2023	Mamba	Выборочное состояние (SSM)	Специализированная реализация



Архитектура трансформера (26)

Трансформер состоит из двух основных компонентов:

Энкодер:

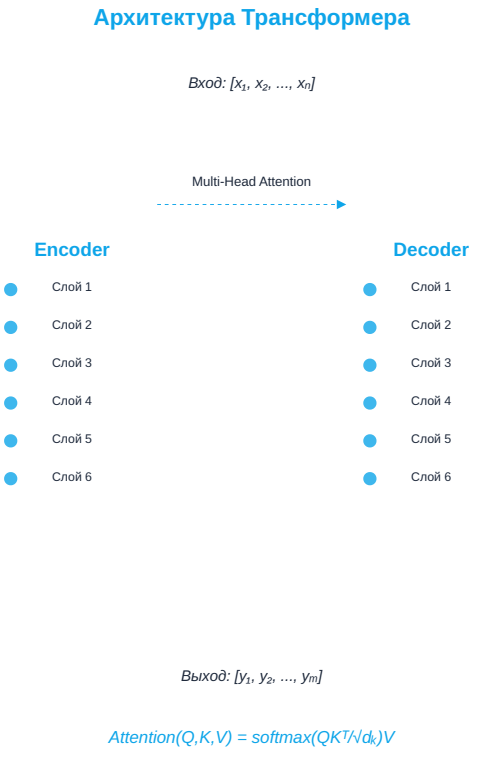
- Стек из N идентичных слоёв
- Каждый слой: много-головое само-внимание + полносвязная сеть с пропусками
- Обработка входной последовательности $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$

Декодер:

- Стек из N идентичных слоёв
- Каждый слой: маскированное внимание + внимание к выходам энкодера + полносвязная сеть
- Авторегрессивная генерация выхода $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_m]$

Механизм внимания:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$



Mamba: линейная сложность для длинных последовательностей (27)

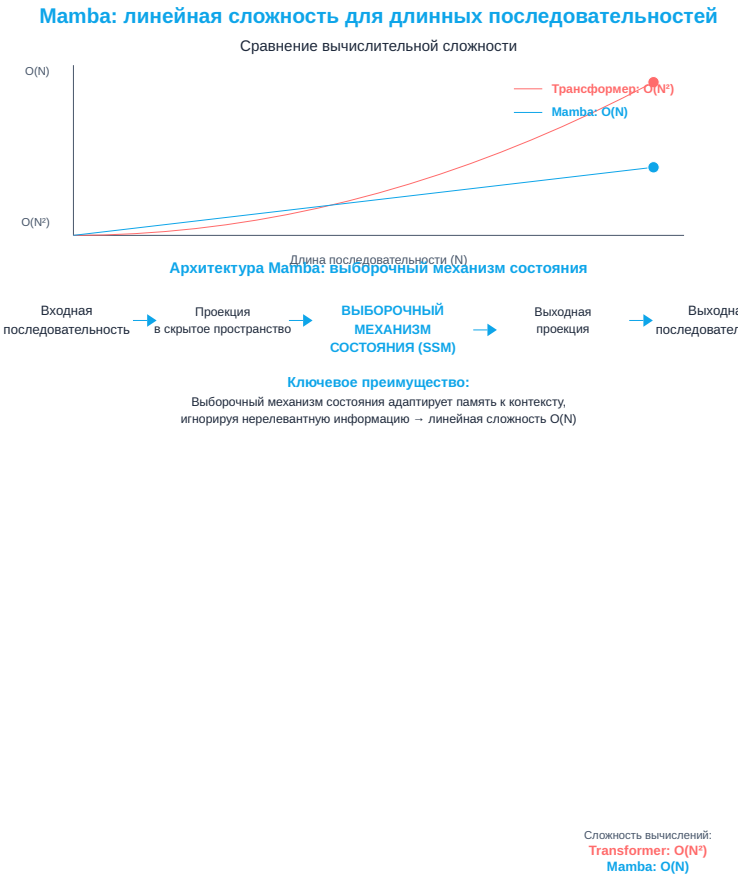
Проблема трансформеров: квадратичная вычислительная сложность $O(N^2)$.
Решение Mamba: выборочный механизм состояния (Selective State Space Model):

$$\mathbf{h}_t = \bar{A}(\Delta_t)\mathbf{h}_{t-1} + \bar{B}(\Delta_t)\mathbf{x}_t$$

$$\mathbf{y}_t = \bar{C}(\Delta_t)\mathbf{h}_t + \bar{D}(\Delta_t)\mathbf{x}_t$$

Ключевые свойства:
- **Адаптивная память:** динамическое решение о сохранении/забывании информации
- **Игнорирование нерелевантного:** подавление шума без дополнительных вычислений
- **Линейная сложность** $O(N)$**:** обработка последовательностей любой длины

Результат: эффективная обработка контекстов длиной 100 000+ токенов.



KAN: Kolmogorov-Arnold Networks (28)

Альтернатива многослойным перцептронам на основе теоремы Колмогорова-Арнольда:

Теорема: любая многомерная непрерывная функция может быть представлена как суперпозиция функций одной переменной:

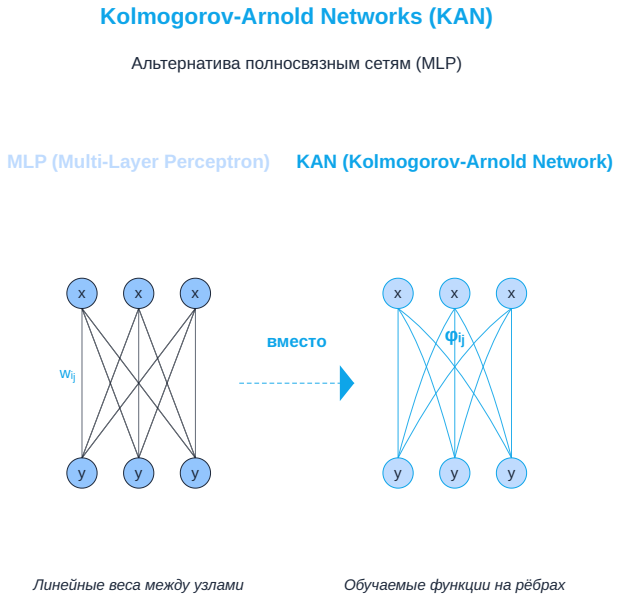
$$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$$

Архитектурное отличие от MLP:

- MLP: обучаемые веса + фиксированные нелинейности
- KAN: фиксированные веса + обучаемые нелинейности на рёбрах

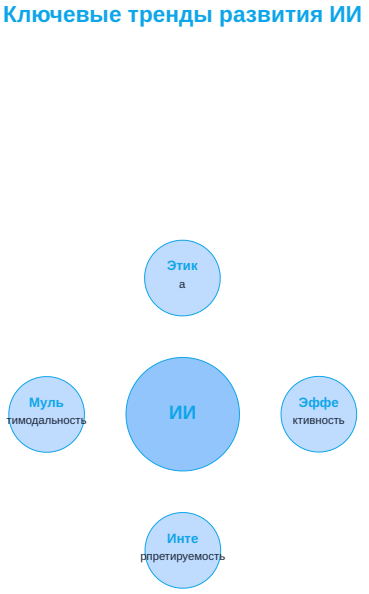
Преимущества:

- Интерпретируемость: каждая функция ϕ_{ij} соответствует влиянию признака
- Эффективность: меньше параметров при той же выразительности



Будущее архитектур ИИ: четыре ключевых направления (29)

1. **Вычислительная эффективность**
- Снижение сложности с $O(N^2)$ до $O(N)$ (Mamba, SSM)
- Оптимизация для периферийных устройств (edge AI)
2. **Интерпретируемость и объяснимость**
- Отказ от «чёрных ящиков» (KAN, attention visualization)
- Формальная верификация поведения моделей
3. **Мультимодальность**
- Единые архитектуры для текста, изображений, аудио, видео
- Кросс-модальное рассуждение и перенос знаний
4. **Этическая ответственность**
- Снижение предвзятости в данных и алгоритмах
- Прозрачность и подотчётность систем ИИ



Ключевые выводы (30)

1. **Иерархия понятий:** ИИ ⊃ интеллектуальные системы ⊃ (символьные системы ∪ системы машинного обучения)
2. **Два фундаментальных подхода:**
- Дедуктивный (экспертные системы): знания → вывод
- Индуктивный (нейросети): данные → обобщение
3. **Эволюция архитектур:** от рекуррентности к вниманию к выборочным состояниям
4. **Будущее ИИ:** инструмент расширения человеческих возможностей при условии этического проектирования и прозрачности.