Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Московский политехнический университет» **(МОСКОВСКИЙ ПОЛИТЕХ)**

Методическое пособие по базовым понятиям, связанным с ИИ для стажеров в различные компании

Москва, 2025

Составители:

1. Захаров А.С., студент МПУ
2. Ким А.А., студент МПУ
3. Угрюмова С.А., студент МПУ
4. Винокуров А.Д., студент МПУ
5. Муратов К.Г., студент МПУ
6. Домрачева А.А., студент МПУ
7. Ионова К.А., студент МПУ
8. Полева М.А., студент МПУ
9. Нармухомедов А.И., студент МПУ
10. Елин Н.С., студент МПУ
11. Пучков А.А., студент МПУ
12. Тигля К.С., студент МПУ

Современный бизнес все активнее использует технологии искусственного интеллекта (ИИ), которые трансформируют подходы к решению задач, анализу данных и автоматизации процессов. Для эффективной работы в новой цифровой реальности необходимо понимать базовые принципы, лежащие в основе этих технологий.

Данное методическое пособие предназначено для стажеров, которые делают первые шаги в своей карьере и стремятся разобраться в фундаментальных понятиях, связанных с ИИ. Мы рассмотрим ключевую терминологию, основные типы и задачи ИИ, а также примеры его практического применения в различных отраслях. Цель этого пособия — дать вам уверенный старт, позволив говорить на одном языке с техническими специалистами и осознанно подходить к использованию ИИ-инструментов в своей работе.

Содержание

Определение ИИ и машинного обучения

Искусственный Интеллект (Artificial Intelligence, AI)

Простое и понятное определение:
Искусственный интеллект (Artificial Intelligence, AI) — это обширная область компьютерных наук, целью которой является создание машин и систем, способных выполнять задачи, требующие человеческого интеллекта.

Ключевая мысль: ИИ — это **цель** или **амбиция**, а не один конкретный метод. Это зонтичный термин, который охватывает множество подходов.

Детальное объяснение ИИ:

1. **Что такое "человеческий интеллект" в этом контексте?**
   Это не обязательно означает полное сознание или самосознание. Речь идет о конкретных когнитивных функциях:
   - **Обучение (Learning):** Способность приобретать новые знания и навыки.
   - **Рассуждение (Reasoning):** Способность логически выводить заключения из имеющейся информации.
   - **Решение проблем (Problem-solving):** Способность находить пути к достижению цели, преодолевая препятствия.
   - **Восприятие (Perception):** Способность интерпретировать информацию из окружающего мира (через зрение, звук и т.д.).
   - **Понимание языка (Language understanding):** Способность не только распознавать слова, но и понимать их смысл, контекст и интенции.
2. **Подходы к созданию ИИ:**
   - **Символический ИИ (Symbolic AI) или "Правила сверху вниз":** Классический подход, который доминировал на заре ИИ. Программист вручную прописывает множество жестких правил и логических конструкций ("если... то..."). Например, экспертная система для медицинской диагностики, где закодированы знания врачей.
     - **Плюсы:** Прозрачность, предсказуемость.
     - **Минусы:** Негибкость, невозможность охватить все ситуации, требует огромных усилий по формализации знаний.
   - **Машинное обучение (Machine Learning) или "Данные снизу вверх":** Современный доминирующий подход. Вместо того чтобы программировать правила, мы позволяем алгоритму самому найти эти правила и закономерности в большом количестве данных.
     - **Плюсы:** Гибкость, способность решать сложные задачи (распознавание образов), адаптивность.

        - **Минусы:** Требует много данных, часто является "черным ящиком".
3. **Уровни ИИ (по классификации):**
    - **Слабый ИИ (Artificial Narrow Intelligence, ANI):** Это весь ИИ, который существует сегодня. Система превосходно решает **одну конкретную задачу**. Примеры: распознавание лиц в Facebook, голосовой помощник Siri, алгоритм рекомендаций Netflix, беспилотный автомобиль. Он не обладает сознанием или общим интеллектом.
    - **Общий ИИ (Artificial General Intelligence, AGI):** Гипотетический ИИ, который по своим интеллектуальным способностям был бы сравним с человеком. Он мог бы научиться выполнять **любую** интеллектуальную задачу, которую может выполнить человек. Это цель многих исследований, но она еще не достигнута.
    - **Суперинтеллект (Artificial Superintelligence, ASI):** Гипотетический ИИ, который превосходит человеческий интеллект во всех сферах. Это сюжет для научной фантастики.

Машинное обучение (МО) — Ключевой Подход к Достижению ИИ

Простое и понятное определение:
Машинное обучение (Machine Learning, ML) — это **подмножество** искусственного интеллекта. Это техника или метод, который позволяет компьютерам **учиться на данных** без явного программирования на выполнение конкретной задачи.

Ключевая мысль: Если ИИ — это цель "создать разумную машину", то МО — это **инструмент** для достижения этой цели, основанный на данных.

Детальное объяснение Машинного Обучения:

1. **Ключевое отличие от традиционного программирования:**
    - **Традиционное программирование:** Программист пишет правила (Rules), компьютер применяет их к входным данным (Input) и выдает результат (Output).
        - *Пример:* Вы пишете код: "Если в тексте есть слова 'злой', 'плохой', 'ужасный' -> это негативный отзыв". Компьютер просто выполняет эту инструкцию.
    - **Машинное обучение:** Мы подаем на вход компьютеру данные (Input) и желаемые результаты (Output), и он **сам находит** правила (Rules), которые связывают одно с другим.

        - *Пример:* Вы подаете тысячи отзывов, каждый из которых помечен как "позитивный" или "негативный". Алгоритм МО анализирует их и сам обнаруживает, что слова "отлично", "прекрасно", "рекомендую" чаще встречаются в позитивных отзывах, а "ужас", "кошмар", "разочарован" — в негативных. Он сам выучил "правила" классификации.
2. **Основные парадигмы (типы) машинного обучения:**
    - **Обучение с учителем (Supervised Learning):**
        - **Суть:** Алгоритм обучается на размеченных данных. Каждому примеру на входе соответствует правильный ответ (метка) на выходе.
        - **Задачи:**
            - **Классификация (Classification):** Отнесение объекта к одному из классов. *Примеры:* Определение спама в email, распознавание рукописных цифр, диагностика заболевания по снимку.
            - **Регрессия (Regression):** Прогнозирование численного значения. *Примеры:* Предсказание цены дома по его параметрам, прогноз продаж на следующий месяц.
    - **Обучение без учителя (Unsupervised Learning):**
        - **Суть:** Алгоритм работает с данными, у которых нет меток. Его задача — найти скрытые структуры, закономерности или сгруппировать данные.
        - **Задачи:**
            - **Кластеризация (Clustering):** Группировка объектов в "кластеры" так, чтобы объекты внутри одного кластера были похожи. *Примеры:* Сегментация клиентов для маркетинга, группировка новостных статей по темам.
            - **Понижение размерности (Dimensionality Reduction):** Упрощение данных без потери важной информации. Помогает визуализировать сложные данные.
    - **Обучение с подкреплением (Reinforcement Learning):**
        - **Суть:** Агент (алгоритм) учится, взаимодействуя со средой. Он совершает действия, за которые получает "вознаграждения" (положительные или отрицательные). Его цель — максимизировать cumulative reward.
        - **Задачи:** *Примеры:* Обучение робота ходьбе, игра в шахматы или го (AlphaGo), беспилотные автомобили.
3. **Что такое "модель" в машинном обучении?**
   Это итоговый "продукт" процесса обучения. По сути, это математическая функция (набор параметров и правил), которая была настроена (обучена) на данных для выполнения конкретной задачи. Когда вы подаете на вход

модели новые, никогда не виденные ею данные, она выдает прогноз (предсказание).

Связь и Иерархия: ИИ vs. Машинное обучение

Представьте себе эту связь в виде матрешки или схемы:

- **Искусственный Интеллект (AI)** — Самая большая, всеобъемлющая область.
    - **Машинное обучение (ML)** — Ключевое подмножество ИИ, наиболее популярный и эффективный на сегодня метод.
        - **Глубокое обучение (Deep Learning)** — Подмножество машинного обучения, основанное на использовании искусственных нейронных сетей с множеством слоев ("глубоких").
            - **Нейронные сети (Neural Networks)** — Конкретная архитектура алгоритмов, вдохновленная строением человеческого мозга, которая является основой для глубокого обучения.

Аналогия:

- **ИИ** — это цель "построить транспортное средство, которое сможет перевозить людей".
- **Машинное обучение** — это конкретный инженерный подход, например, "использовать двигатель внутреннего сгорания".
- **Глубокое обучение** — это передовая технология в рамках этого подхода, например, "турбированный двигатель с непосредственным впрыском топлива".

Сферы машинного обучения

Существует три основные парадигмы, которые отличаются по типу данных и способу "обучения" алгоритма. Вот их обзор, а ниже — детальное объяснение каждой.

1. Обучение с учителем (Supervised Learning)

Ключевая идея: Обучение на **размеченных данных**. Это означает, что каждому объекту в обучающей выборке соответствует "правильный ответ" (метка). Алгоритм изучает взаимосвязь между входными данными и этими метками, чтобы в дальнейшем предсказывать их для новых, никогда не виденных данных.

Аналогия: Ученик (алгоритм), который учится решать задачи, имея под рукой учебник с условиями задач и **готовыми ответами** (размеченные данные). Он сверяет свои решения с правильными, понимает свои ошибки и учится.

Основные задачи в рамках Обучения с учителем:

а) Классификация (Classification)

- **Цель:** Отнести объект к одной из **предопределенных категорий**.
- **Входные данные:** Признаки объекта.
- **Выходные данные:** Дискретная метка класса (категория).
- **Примеры:**
    - **Бинарная классификация (2 класса):**
        - *Спам-фильтр:* Вход - текст email, Выход - "спам" или "не спам".
        - *Медицинская диагностика:* Вход - симптомы и анализы, Выход - "болен" или "здоров".
    - **Многоклассовая классификация (>2 классов):**
        - *Распознавание рукописных цифр:* Вход - изображение цифры, Выход - "0", "1", ..., "9".
        - *Классификация изображений:* Вход - фотография, Выход - "кошка", "собака", "автомобиль".
- **Популярные алгоритмы:** Логистическая регрессия, Решающие деревья, Метод опорных векторов (SVM), Наивный байесовский классификатор, Нейронные сети.

б) Регрессия (Regression)

- **Цель:** Предсказать **непрерывное численное значение**.
- **Входные данные:** Признаки объекта.
- **Выходные данные:** Число.
- **Примеры:**
    - *Предсказание цены на недвижимость:* Вход - площадь, район, количество комнат, Выход - цена (в рублях/долларах).
    - *Прогноз продаж:* Вход - исторические данные о продажах, сезонность, маркетинговый бюджет, Выход - объем продаж на следующий месяц.
    - *Предсказание температуры:* Вход - данные о погоде за последние дни, Выход - температура завтра.
- **Популярные алгоритмы:** Линейная регрессия, Полиномиальная регрессия, Решающие деревья для регрессии, Гребневая регрессия (Ridge).

2. Обучение без учителя (Unsupervised Learning)

Ключевая идея: Обучение на **данных без меток**. Алгоритм не получает "правильных ответов". Его задача — самостоятельно найти **скрытые структуры, закономерности или аномалии** в данных.

Аналогия: Дать ученику (алгоритму) кучу разноцветных и разновысоких кубиков (данные без меток) и попросить разложить их по каким-то своим принципам. Он может сгруппировать их по цвету, по размеру или найти какую-то другую, неочевидную нам закономерность.

Основные задачи в рамках Обучения без учителя:

а) Кластеризация (Clustering)

- **Цель:** Разбить данные на группы (**кластеры**) таким образом, чтобы объекты внутри одного кластера были **максимально похожи** друг на друга, а объекты из разных кластеров — **максимально отличались**.
- **Примеры:**
    - *Сегментация клиентов:* Группировка клиентов интернет-магазина на основе их purchasing behavior (частота покупок, средний чек, категории товаров). Это помогает разработать персонализированные маркетинговые стратегии для каждого сегмента.
    - *Группировка документов:* Автоматическое объединение новостных статей по темам без заранее заданных категорий.
    - *Анализ в биоинформатике:* Группировка генов с похожими функциями.

б) Понижение размерности (Dimensionality Reduction)

- **Цель:** Уменьшить количество признаков в данных, сохранив при этом как можно больше полезной информации. Часто используется для визуализации или для упрощения данных перед обучением другой модели.
- **Примеры:**
    - *Визуализация многомерных данных:* У вас есть данные о клиентах с 50 признаками (возраст, доход, расходы и т.д.). Человек не может воспринять 50-мерное пространство. Алгоритм понижения размерности может сжать эти 50 признаков до 2-3 главных, чтобы можно было построить график и увидеть группы клиентов.
    - *Сжатие данных:* Как JPEG для изображений, но для произвольных данных.
- **Популярные алгоритмы:** Метод главных компонент (PCA), t-SNE, UMAP.

в) Обнаружение аномалий (Anomaly Detection)

- **Цель:** Найти объекты, которые сильно отличаются от большинства данных ("выбросы").
- **Примеры:** Обнаружение мошеннических операций с кредитными картами, выявление неисправного оборудования по данным с датчиков, поиск сетевых атак.

3. Обучение с подкреплением (Reinforcement Learning)

Ключевая идея: Агент (алгоритм) учится взаимодействовать со **средой**, чтобы максимизировать cumulative reward (суммарное вознаграждение). Агент не получает готовых примеров "что делать". Он учится методом проб и ошибок.

Ключевые понятия:

- **Агент (Agent):** Само обучающаяся система (например, программа для игры в шахматы).
- **Среда (Environment):** Мир, в котором существует агент (шахматная доска).
- **Действие (Action):** То, что агент может делать (ходить фигурой).
- **Состояние (State):** Текущая ситуация в среде (позиция фигур на доске).
- **Вознаграждение (Reward):** Числовая обратная связь от среды за выполненное действие (например, +1 за выигрыш, -1 за проигрыш, 0 за остальные ходы).

Аналогия: Дрессировка собаки. Собака (агент) в среде (квартира). Она выполняет действие (садится). Дрессировщик (среда) дает ей лакомство (положительное вознаграждение) или не дает (отрицательное). Собака не знала

изначально команду "сидеть", но она методом проб и ошибок поняла, какая последовательность действий приводит к максимальному вознаграждению.

Примеры:

- **Игры:** AlphaGo и AlphaZero (игра в Го и шахматы), боты для Dota 2. Агент учился, играя миллионы раз против себя самого.
- **Робототехника:** Обучение робота ходьбе. Вознаграждение дается за движение вперед без падения.
- **Беспилотные автомобили:** Агент получает положительное вознаграждение за безопасное и плавное вождение, и отрицательное — за нарушения и аварии.
- **Управление ресурсами:** Оптимизация расходов энергии в дата-центре.

Популярные алгоритмы: Q-Learning, Deep Q-Network (DQN), Policy Gradients.

Другие важные сферы (гибридные)

4. Частичное обучение (Semi-supervised Learning)

- **Идея:** Используется когда есть **немного размеченных данных** и **очень много неразмеченных**. Размеченные данные помогают задать общее направление, а неразмеченные — уточнить структуру данных. Это очень распространенный сценарий на практике, так как разметка данных — дорогой и трудоемкий процесс.

5. Самообучение (Self-supervised Learning)

- **Идея:** Это мощный подвид обучения без учителя, где модель сама создает для себя "учителя". Например, она может скрывать часть данных (например, кусок изображения или слова в тексте) и пытаться предсказать скрытую часть по оставшемуся контексту. Так обучаются современные большие языковые модели (как GPT).

6. Активное обучение (Active Learning)

- **Идея:** Модель сама решает, какие данные из неразмеченного пула ей наиболее полезно было бы разметить экспертом. Она "задает вопросы", тем самым уменьшая общие затраты на разметку данных.

Сводная таблица

Сфера	Данные для обучения	Цель	Ключевые задачи
С учителем	Размеченные (есть "правильный ответ")	Предсказать метку для новых данных	Классификация, Регрессия
Без учителя	Неразмеченные (нет ответа)	Найти скрытую структуру	Кластеризация, Понижение размерности, Обнаружение аномалий
С подкреплением	Взаимодействие со средой	Максимизировать cumulative reward	Принятие последовательных решений, Управление

Нейросети и языковые модели

Часть 1: Нейронные Сети (Neural Networks)

Простое определение: Нейронная сеть — это математическая модель, вдохновленная строением и работой биологического мозга, предназначенная для распознавания сложных паттернов в данных.

Детальное объяснение: Как устроена и работает нейронная сеть

1. Биологический прототип: Нейрон в мозге

- У нас в мозге ~86 миллиардов нейронов.
- Каждый нейрон:
    - Получает электрические сигналы от других нейронов через **дендриты**.
    - Обрабатывает их в **теле клетки**.
    - Если суммарный сигнал превышает некоторый порог, нейрон "активируется" и передает сигнал дальше через **аксон** к другим нейронам через **синапсы**.

2. Искусственный нейрон (Перцептрон)
Это упрощенная математическая модель биологического нейрона.

- **Входы (x_1, x_2, ...):** Это данные (например, пиксели изображения, слова, числа). Каждому входу соответствует **вес (w_1, w_2, ...)**. Вес — это число, показывающее "важность" данного входа. Чем вес больше, тем сильнее данный вход влияет на результат.
- **Сумматор:** Нейрон вычисляет **взвешенную сумму** входов: $z = (w_1*x_1) + (w_2*x_2) + ... + b$.
- **Смещение (Bias, b):** Это дополнительный параметр, который позволяет смещать порог активации. Это как "константа" в линейном уравнении.
- **Функция активации (Activation Function, f):** Полученная сумма z пропускается через нелинейную функцию. Именно она придает сети способность обучаться сложным, нелинейным зависимостям.
    - **Зачем нужна?** Без нее вся сеть была бы просто одним большим линейным преобразованием, неспособным научиться чему-либо сложному.
    - **Примеры функций:** ReLU (выпрямитель), Сигмоида, Гиперболический тангенс.

Выход нейрона: $y = f(z)$

Аналогия: Представьте, что вы решаете, идти ли вам на вечеринку.

- **Входы:** Погода (x_1), наличие друзей (x_2), усталость (x_3).

- **Веса:** Для вас наличие друзей (w_2) гораздо важнее, чем погода (w_1).
- **Смещение:** Ваше общее отношение к вечеринкам (вы интроверт или экстраверт).
- **Функция активации:** Логика принятия решения. "Если общий 'уровень желания' > 6, то я иду".
- **Выход:** "Да" или "Нет".

3. От нейрона к сети: Многослойный перцептрон (MLP)

Один нейрон — слабый вычислитель. Мощь возникает, когда мы соединяем тысячи и миллионы нейронов в **слои**.

- **Входной слой (Input Layer):** Получает исходные данные (например, 784 нейрона для изображения 28x28 пикселей).
- **Скрытые слои (Hidden Layers):** Это "мозг" сети. Они находятся между входом и выходом, выполняют основную вычислительную работу по извлечению и преобразованию признаков. Чем больше слоев, тем **"глубже"** сеть (отсюда термин **Глубокое обучение**).
- **Выходной слой (Output Layer):** Выдает окончательный результат. Его структура зависит от задачи:
    - **Классификация:** Часто использует функцию **Softmax**, которая преобразует выходы в вероятности (сумма всех выходов = 1). Например, для распознавания цифр: на выходе 10 нейронов, каждый показывает вероятность того, что это цифра 0, 1, ..., 9.
    - **Регрессия:** Один нейрон с линейной функцией активации (предсказывает число).

Процесс работы (Прямое распространение, Forward Pass):
Данные последовательно проходят от входного слоя через все скрытые слои к выходному. На каждом нейроне происходит операция: взвешенная сумма -> применение функции активации.

4. Как нейросеть учится? (Обучение)

Это самый важный и сложный аспект. Обучение — это **настройка весов и смещений** так, чтобы сеть выдавала максимально точные результаты.

- **Функция потерь (Loss Function):** Это мера того, "насколько сильно наша сеть ошиблась". Сравнивает предсказание сети с правильным ответом. Например, "Среднеквадратичная ошибка" для регрессии или "Перекрестная энтропия" для классификации.
- **Цель обучения:** Минимизировать значение функции потерь. Минимум функции потерь = наилучшие предсказания.

Метод: Алгоритм обратного распространения ошибки (Backpropagation) и Градиентный спуск (Gradient Descent)
Это два столпа обучения нейросетей.

1. **Прямой проход:** Подаем на вход данные, получаем предсказание, вычисляем ошибку (функцию потерь).
2. **Обратный проход (Backpropagation):** Это умный алгоритм, который вычисляет **градиент** функции потерь. Градиент — это вектор, показывающий направление **наискорейшего роста** функции. Нам нужно двигаться в обратном направлении.
    - **Как работает?** С помощью цепного правила из математического анализа, алгоритм эффективно вычисляет, как каждый вес в сети (даже в самых первых слоях) влияет на итоговую ошибку. Он "распространяет" ошибку от выхода назад по сети.
3. **Градиентный спуск (Gradient Descent):** Получив градиент (направление, в котором нужно двигаться, чтобы УВЕЛИЧИТЬ ошибку), мы делаем маленький шаг в **обратную сторону**, чтобы ошибку **УМЕНЬШИТЬ**.
    - **Скорость обучения (Learning Rate):** Размер этого шага. Слишком большой шаг — "перепрыгнем" минимум, слишком маленький — обучение будет очень медленным.

Процесс повторяется тысячи/millions раз: Данные -> Прямой проход -> Вычисление ошибки -> Обратный проход (вычисление градиента) -> Корректировка весов (шаг градиентного спуска). И так по кругу, пока ошибка не станет приемлемо малой.

Часть 2: Языковые Модели (Language Models, LM)

Простое определение: Языковая модель — это модель, которая изучает вероятностное распределение над последовательностями слов. Проще говоря, она **предсказывает вероятность появления следующего слова (или слова) в последовательности**.

Ключевая идея: Языковая модель "понимает" язык, улавливая статистические закономерности: какие слова часто встречаются вместе, какие конструкции грамматически верны, каков стиль текста.

Детальное объяснение: Эволюция и принципы языковых моделей

1. Классические подходы (до нейросетей)

- **N-gram модели:**
    - **Идея:** Предсказывают следующее слово на основе предыдущих (N-1) слов.
    - **Пример (Триграмма, 3-gram):** "я люблю ___". Модель ищет в своем тренировочном корпусе (огромной коллекции текстов), что чаще всего стоит после "люблю": "тебя", "кофе", "гулять". И выбирает наиболее вероятный вариант.

- **Проблемы:** Не может учитывать длинные контексты (ограничение N). Не может понять смысл, только статистику. Столкнувшись с новой фразой, которой не было в корпусе, дает нулевую вероятность ("проклятие размерности").

2. Нейросетевые языковые модели (эра Глубокого обучения)

С появлением нейросетей языковое моделирование совершило гигантский скачок.

- **Word Embeddings (Векторные представления слов):**
  - **Идея:** Каждому слову ставится в соответствие не индекс, а **вектор** (набор чисел) в многомерном пространстве (например, 300 чисел). Это плотное, распределенное представление.
  - **Ключевое свойство:** Семантически близкие слова имеют близкие векторы. Знаменитый пример: vector("король") - vector("мужчина") + vector("женщина") ≈ vector("королева").
  - **Примеры алгоритмов:** Word2Vec, GloVe.
- **Рекуррентные нейросети (RNN) и LSTM:**
  - **Идея:** Эти архитектуры были специально разработаны для работы с **последовательностями**. У них есть "память" (скрытое состояние), которая передается от одного элемента последовательности к другому.
  - **Как работает LM на RNN:** Сеть по очереди принимает слова предложения. Для каждого слова она использует свою "память" о предыдущих словах, чтобы предсказать следующее. LSTM (Long Short-Term Memory) — усовершенствованная RNN, которая лучше запоминает долгосрочные зависимости.

3. Трансформеры и современные Большие Языковые Модели (LLM)

Это революционная архитектура, лежащая в основе GPT, BERT, T5 и других современных моделей.

- **Ключевое новшество: Механизм Внимания (Attention Mechanism)**
  - **Идея:** Позволяет модели при обработке каждого слова "взглянуть" на **любое другое слово** в предложении (или даже во всем тексте) и решить, насколько оно важно для понимания текущего слова.
  - **Аналогия:** Читая слово "её" в предложении "Маша доела шоколадку, потому что **она** была вкусной", вы автоматически обращаете внимание на слово "шоколадка", чтобы понять, что "она" относится к ней. Механизм внимания делает то же самое вычислительно.
  - **Самовнимание (Self-Attention):** Разновидность, где запрос, ключ и значение берутся из одного и того же набора данных (из одного предложения).

- **Архитектура Трансформера:**
    - **Кодер (Encoder) и Декодер (Decoder):** Изначально Трансформер состоял из этих двух частей.
    - **Кодер** (используется в BERT): Читает весь текст сразу и создает его "глубинное представление". Хорош для задач понимания: классификация текста, извлечение сущностей, вопрос-ответ.
    - **Декодер** (используется в GPT): Генерирует текст последовательно, слово за словом. На каждом шаге он видит только предыдущие слова (маскирование будущих). Идеально для **генерации текста**.
    - **Большие Языковые Модели (LLM)**, такие как GPT, — это **декодерные** трансформеры огромного размера (миллиарды параметров), обученные на колоссальных объемах текстовых данных.

Как обучаются современные LLM (например, GPT)?

1. **Этап 1: Самообучение (Self-supervised Learning / Дообучение без учителя).**
    - **Задача:** Предсказание следующего слова.
    - **Процесс:** Модели подается огромный текст из интернета (книги, статьи, код). Ее задача — предсказать следующее слово в этом тексте. Ошибка вычисляется между ее предсказанием и реальным следующим словом. Через обратное распространение ошибки настраиваются веса.
    - **Результат:** После этого этапа модель уже "знает" язык: грамматику, синтаксис, факты, стили. Она является мощной **базовой моделью (Foundation Model)**.
2. **Этап 2: Дообучение с учителем (Supervised Fine-Tuning, SFT).**
    - **Цель:** Научить модель следовать инструкциям и вести диалог.
    - **Процесс:** Модель обучают на большом количестве примеров "вопрос-ответ", созданных людьми (анантаторами). Например: "Пользователь: Напиши email. Ассистент: Конечно, вот пример...".

3. **Этап 3: Обучение с подкреплением с помощью человеческой обратной связи (RLHF).**
    - **Цель:** Сделать ответы модели более полезными, честными и безопасными.
    - **Процесс:**
        - Модели задают множество вопросов, она генерирует несколько вариантов ответов.
        - Человек-анантатор ранжирует эти ответы от лучшего к худшему.
        - На основе этих рейтингов тренируется **модель вознаграждения (Reward Model)**, которая учится предсказывать, какой ответ понравится человеку.
        - Исходная языковая модель далее дообучается с помощью RL, чтобы максимизировать "вознаграждение" от этой модели. То есть, она учится генерировать такие ответы, которые получили бы высокий рейтинг.

Связь между Нейросетями и Языковыми Моделями

Языковая модель — это ЗАДАЧА. Задача предсказать следующее слово.
Нейросеть (в частности, Трансформер) — это ИНСТРУМЕНТ. Архитектура, которая решает эту задачу.

- **Современные Языковые Модели — это нейросети.** А именно, очень большие нейросети на архитектуре Трансформера.
- **Нейросети используются не только для языкового моделирования.** Они решают задачи компьютерного зрения (сверточные сети CNN), обработки звука (RNN) и многое другое.

Виды нейросетей

Архитектура нейросети определяется тем, как соединены её нейроны. Разные архитектуры подходят для разных типов данных.

1. Полносвязные нейронные сети (Fully Connected / Dense Networks):
    - Строение: Каждый нейрон в слое соединён с каждым нейроном в следующем слое.
    - Применение: Простые задачи классификации и регрессии, где нет явной пространственной или временной структуры (например, прогноз на основе табличных данных).
2. Свёрточные нейронные сети (Convolutional Neural Networks, CNN):
    - Строение: Используют свёрточные слои, которые применяют «фильтры» (ядра) к небольшим областям входных данных (например, к группам пикселей). Это позволяет эффективно находить локальные паттерны: края, углы, текстуры, а затем и более сложные объекты.
    - Ключевая идея: Инвариантность к перемещению и иерархическое обучение признаков.
    - Применение: Преимущественно для работы с изображениями (распознавание, классификация, сегментация), но также и с видео, и даже с звуком.
3. Рекуррентные нейронные сети (Recurrent Neural Networks, RNN):
    - Строение: Имеют «память». У нейронов есть петли, позволяющие информации сохраняться. Они обрабатывают данные последовательно (одно слово за другим) и учитывают предыдущие состояния при обработке текущего.
    - Проблема: Классические RNN страдают от проблемы «затухающего градиента» и плохо запоминают долгосрочные зависимости.
    - Развитие: LSTM (Long Short-Term Memory) и GRU (Gated Recurrent Unit) — более сложные типы RNN, которые решают проблему долгосрочной памяти с помощью специальных «вентилей».
    - Применение: Ранее были основой для обработки текста, машинного перевода, но сейчас в основном вытеснены Трансформерами.
4. Трансформеры (Transformers):
    - Строение: Основаны исключительно на механизме внимания, без рекуррентных связей. Это позволяет им обрабатывать все элементы последовательности (все слова в предложении) параллельно, что значительно ускоряет обучение.
    - Преимущество: Высокая эффективность, масштабируемость и способность улавливать сложные зависимости в данных, независимо от расстояния между элементами.
    - Применение: Фактически стал стандартом для всех современных задач обработки естественного языка (NLP): языковые модели (GPT,

BERT), перевод, суммаризация. Начинает применяться и для изображений (Vision Transformers, ViT).

5. Генеративно-состязательные сети (Generative Adversarial Networks, GAN):
    - Строение: Состоят из двух сетей, которые соревнуются друг с другом в игре («генератор» и «дискриминатор»).
        - Генератор: Создаёт поддельные данные (например, изображения кошек) из случайного шума.
        - Дискриминатор: Получает на вход и реальные изображения кошек, и сгенерированные. Его задача — определить, какие из них настоящие.
    - Процесс: В процессе соревнования генератор учится создавать всё более реалистичные данные, а дискриминатор — всё лучше их распознавать.
    - Применение: Генерация фотореалистичных изображений, стилизация, увеличение разрешения (супер-резолюция).
6. Автокодировщики (Autoencoders):
    - Строение: Состоят из двух частей — энкодера, который сжимает входные данные в компактное представление (код), и декодера, который пытается восстановить исходные данные из этого кода.
    - Цель: Научиться эффективному сжатию данных (понижению размерности) или очистке данных от шума.
    - Применение: Удаление шума с изображений, поиск аномалий, системы рекомендаций.

Что такое промт, базовое взаимодействие с ИИ

Промт (Prompt) — инструкция или запрос

Определение: Промт — это текст, который пользователь вводит в языковую модель (например, в ChatGPT), чтобы получить желаемый ответ.
Это инструкция, вопрос, контекст или команда, определяющая, что именно должна сделать модель.

Важность: Качество и детализация промта напрямую влияют на качество и релевантность ответа ИИ. Промтинг — это искусство эффективного общения с искусственным интеллектом.

Базовое взаимодействие с ИИ: принципы и техники

1. Будьте конкретны и ясны.
    - *Плохо:* «Напиши про собак». (Слишком широко. Про что именно? Про породы? Уход? Питание?)
    - *Хорошо:* «Напиши информационный пост для блога о собаководах, посвящённый трём самым популярным породам собак для квартиры: их характеру, уровню активности и основным правилам ухода. Объём — 500 слов.»
2. Задавайте роль (Role-Playing). Это мощнейший приём.
    - *Пример:* «Ты — опытный копирайтер с 10-летним стажем, специализирующийся на написании продающих текстов для IT-стартапов. Напиши цепляющий заголовок и первый абзац для лендинга нового мобильного приложения для учёта финансов.»
3. Предоставляйте контекст. Чем больше релевантной информации вы дадите, тем точнее будет ответ.
    - *Пример:* «Я готовлю презентацию для инвесторов в сфере возобновляемой энергетики. Моя аудитория — люди 50+ с техническим образованием. Цель — убедить их вложиться в строительство солнечной электростанции. Напиши три сильных аргумента, которые будут для них убедительны.»
4. Используйте пошаговые инструкции (Chain-of-Thought). Если задача сложная, разбейте её.
    - *Пример:* «Объясни, почему небо голубое. Сначала объясни на научном уровне (рассеяние Рэлея), а затем приведи простое сравнение для понимания ребёнком 7 лет.»

-
5. Показывайте примеры (Few-Shot Prompting). Иногда проще показать, чем объяснить.
    - *Пример:*
        - Промт: «Классифицируй тональность отзывов на фильмы. Вот примеры:
            - Отзыв: "Актерская игра была великолепна, но сюжет предсказуем." -> Тональность: Смешанная
            - Отзыв: "Это самый скучный фильм, который я видел за год." -> Тональность: Негативная
            - Отзыв: "Потрясающая графика и захватывающий сюжет!" -> Тональность: Позитивная
            - Теперь классифицируй этот отзыв: "Декорации красивые, но диалоги слабые." -> »
6. Уточняйте формат вывода.
    - *Пример:* «Перечисли 5 преимуществ удалённой работы. Оформи ответ в виде маркированного списка.»
7. Итеративный процесс: не бойтесь уточнять! Взаимодействие с ИИ — это диалог.
    - Если ответ не устроил, можно:
        - Уточнить: «Перефразируй это короче».
        - Расширить: «Расскажи подробнее о втором пункте».
        - Исправить: «Ты ошибся в факте. На самом деле... Пересмотри свой ответ с учётом этого».

Список литературы

1. Горохов Александр Владимирович, Мартынов Вячеслав Андреевич, Гаврин Виталий Алексеевич ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ // Скиф. 2022. №4 (68).
2. Вахрушева М. А. Искусственный интеллект // Интеллектуальный потенциал XXI века: ступени познания. 2011. №6.
3. Гельдиев Б. А., Хатджиева О. К., Куллыева О. Х., Байрамова С. ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ // Символ науки. 2023. №11-1-2.
4. Рассел С., Норвиг П. *Искусственный интеллект: современный подход*. – М.: Вильямс, 2021. – 1412 с.
5. Гудфеллоу Я., Бенджио И., Курвилль А. *Глубокое обучение*. – М.: ДМК Пресс, 2022. – 652 с.
6. Мерфи К.П. *Машинное обучение: вероятностный подход*. – М.: ДМК Пресс, 2023. – 848 с.
7. Джурафски Д., Мартин Д. *Speech and Language Processing*. – 3rd ed. – Prentice Hall, 2023. – 512 с.
8. Тун Л., Голдберг Й., Мамони Р. *Прикладная обработка естественного языка*. – М.: ДМК Пресс, 2022. – 436 с.
9. Раш А., Вольф Т. *Natural Language Processing with Transformers*. – O'Reilly Media, 2022. – 368 с.
10. Жерон О. *Скраппинг веб-сайтов с помощью Python: сбор данных из современного интернета*. – М.: Питер, 2022. – 688 с.
11. Бхаргава А. *Грокаем алгоритмы*. – М.: Питер, 2022. – 288 с.
12. Мюллер А., Гвидо С. *Введение в машинное обучение с помощью Python*. – М.: ДМК Пресс, 2023. – 480 с.