Контрольные вопросы

1. В чем преимущества гибридных интеллектуальных систем перед системами одного типа?

**1. Повышенная адаптивность и гибкость**

* Гибридные системы могут переключаться между методами в зависимости от задачи: например, использовать правила для логических выводов и ML для обработки больших данных.
* Преимущество: Лучше справляются с разнообразными входами, чем чисто ML-системы или rule-based.

**2. Улучшенная точность и надёжность**

* Комбинируют сильные стороны: ML справляется с паттернами в данных, а экспертные системы — с интерпретируемостью и логикой.
* Преимущество: Снижают ошибки; например, в медицинской диагностике гибридный ИИ минимизирует ложные срабатывания по сравнению с чисто нейронными сетями, которые могут быть "чёрными ящиками".

**3. Лучшая обработка неопределённости и сложности**

* Интегрируют вероятностные модели (из ML) с дедуктивными, позволяя работать с неоднозначными данными.
* Преимущество: Эффективны в динамичных средах, где системы одного типа могут плохо справляться с редкими событиями.

**4. Оптимизация ресурсов и эффективности**

* Гибридные системы могут делегировать задачи: лёгкие вычисления — правилам, тяжёлые — ML.
* Преимущество: Экономят вычислительные ресурсы и время обучения; например, в рекомендательных системах гибрид снижает нагрузку по сравнению с чисто ML-моделями.

**5. Интерпретируемость и этичность**

* Добавляют прозрачность: правила объясняют решения, в отличие от "чёрных ящиков" в глубоких нейронных сетях.
* Преимущество: Повышают доверие пользователей и соответствие регуляциям, где монолитные системы одного типа часто непрозрачны.

1. Какие методы представления знаний наиболее эффективны для экспертных систем?

**Основные методы представления знаний**

**1. Правила (production rules / IF–THEN)**

* Описание: правила вида «если условие — то действие/заключение».
* Сильные стороны: прозрачность, легко читать/редактировать экспертам, простая реализация механизма вывода (forward/backward chaining).
* Ограничения: может разрастаться, плохо моделирует неопределённость и взаимодействия между факторами.
* Когда использовать: диагностические системы, бизнес-правила, системы принятия решений с чёткими условиями.
* Примеры инструментов: системы правил (Drools, CLIPS).

**2. Онтологии и логическое представление (Description Logics / OWL / логическое программирование)**

* Описание: формальные модели домена (классы, свойства, отношения) с логическим выводом.
* Сильные стороны: строгая семантика, поддержка дедукции, интероперабельность, соответствует семантическому вебу.
* Ограничения: сложность моделирования процедурного знания, вычислительная стоимость для очень больших онтологий.
* Когда использовать: сложные предметные области с богатой структурой объектов, интеграция данных, семантический поиск.
* Примеры инструментов: OWL-редакторы, reasoners (Pellet, HermiT).

**3. Байесовские сети и вероятностные графы**

* Описание: графы причинно-следственных зависимостей с вероятностными оценками.
* Сильные стороны: формальная работа с неопределённостью, обучение по данным, качественная интерпретируемость зависимости.
* Ограничения: требуется знание/оценка условных вероятностей; масштабирование при большом числе переменных.
* Когда использовать: медицинская диагностика, прогнозирование, риск-оценка.
* Инструменты: bnlearn, GeNIe, pgmpy.

**4. Нечёткая логика (Fuzzy logic)**

* Описание: представление нечётких/непрерывных понятий (частичная принадлежность).
* Сильные стороны: естественно моделирует человеческие описания («высокая», «умеренно»), простота правил с нечёткими условиями.
* Ограничения: выбор функций принадлежности субъективен, интеграция с вероятностными моделями не всегда тривиальна.
* Когда использовать: управление (контроль, регулирование), экспертные оценки, когда критичны лингвистические переменные.

**5. Сетевые представления / Фреймы / Семантические сети**

* Описание: объекты представлены фреймами (атрибуты, значения), связи между ними — в виде графа.
* Сильные стороны: естественная структуризация знаний об объектах, поддержка наследования и дефолтных значений.
* Ограничения: менее выразительны для сложных логических выводов, требуется аккуратная модель иерархий.
* Когда использовать: представление типов сущностей и их свойств, экспертные базы знаний с объектно-ориентированной природой.

**6. Case-Based Reasoning (CBR) — поиск по прецедентам**

* Описание: хранение прошлых случаев и использование их как шаблонов для новых задач.
* Сильные стороны: хороша, когда есть богатая база реальных историй/примеров; обучение инкрементально.
* Ограничения: требуется механизм извлечения похожих случаев; может не обобщать в новых ситуациях.
* Когда использовать: поддержка решений по прецедентам (юриспруденция, обслуживание, ремонт).

**7. Машинное обучение / нейросети как представление знаний**

* Описание: знания закодированы в параметрах моделей (NN, embeddings).
* Сильные стороны: масштабируемость, способность выявлять сложные нелинейные зависимости.
* Ограничения: низкая объяснимость; для экспертных систем требуется механизмы интерпретации.
* Когда использовать: большие наборы данных, задачи прогнозирования, где прозрачность не критична или обеспечивается отдельными модулями объяснения.

**8. Деревья решений и решающие таблицы**

* Описание: иерархия условий, приводящая к решениям; таблицы удобно структурируют комбинации условий.
* Сильные стороны: понятны, легко переводимы в правила, хороши для бинарных/категориальных задач.
* Ограничения: могут переобучаться (если строятся автоматически), неудобны при многофакторных нечетких условиях.
* Когда использовать: быстрые экспертные решения, когда нужна простая интерпретируемая модель.

1. Как выбрать архитектуру нейронной сети для конкретной задачи классификации?

Выбор архитектуры нейронной сети для задачи классификации зависит от множества факторов: типа данных, сложности задачи, требований к точности и ресурсам. Вот пошаговый подход и основные рекомендации для выбора архитектуры:

1. Анализ типа данных

Табличные данные: обычно подходят многослойные персептроны (MLP, полносвязные сети).

Изображения: хорошо работают сверточные нейронные сети (CNN), которые учитывают пространственную структуру.

Последовательности (текст, временные ряды): эффективны рекуррентные сети (RNN, LSTM, GRU) или трансформеры.

Графы: графовые нейронные сети (GNN).

2. Размер и качество данных

Большие датасеты позволяют использовать более сложные и глубокие архитектуры.

Для малых данных — простые модели или transfer learning (предобученные модели).

3. Цель и требования к модели

Если требуется высокая точность — можно экспериментировать с более глубокими и сложными сетями.

Если важна скорость или ресурсы — выбирать компактные и легковесные архитектуры.

4. Примерная структура сети

Входной слой: соответствует размерности признаков.

Скрытые слои: количество и размер зависят от сложности задачи.

Выходной слой: для классификации с N классами — обычно N нейронов с активацией softmax.

5. Использование готовых архитектур

Для изображений: ResNet, VGG, EfficientNet, MobileNet.

Для текста: BERT, GPT, LSTM с attention.

Можно использовать предобученные модели и дообучать их под свою задачу (transfer learning).

6. Тестирование и настройка

Начинайте с простой архитектуры и постепенно увеличивайте сложность.

Используйте методы регуляризации (Dropout, BatchNorm) чтобы избежать переобучения.

Подбирайте гиперпараметры (число слоев, нейронов, скорость обучения) экспериментально.

7. Оценка результатов

Следите за метриками качества (accuracy, precision, recall, F1 и др.) на валидационной выборке.

1. Какие метрики наиболее важны для оценки качества интеллектуальной системы?

1. Для задач классификации:

Accuracy (Точность) — доля правильно классифицированных объектов. Подходит при сбалансированных классах.

Precision (Точность по положительному классу) — доля правильно предсказанных положительных примеров среди всех предсказанных как положительные.

Recall (Полнота, чувствительность) — доля правильно найденных положительных примеров среди всех истинных положительных.

F1-score — гармоническое среднее precision и recall, полезно при несбалансированных классах.

ROC-AUC (Area Under Curve) — площадь под ROC-кривой, отражает качество классификации при разных порогах.

Confusion Matrix (Матрица ошибок) — показывает распределение предсказаний по классам.

2. Для задач регрессии:

Mean Absolute Error (MAE) — средняя абсолютная ошибка.

Mean Squared Error (MSE) — среднеквадратичная ошибка.

Root Mean Squared Error (RMSE) — корень из MSE, измеряется в тех же единицах, что и целевая переменная.

R-squared (Коэффициент детерминации) — доля объяснённой дисперсии.

3. Для задач кластеризации:

Silhouette Score — измеряет, насколько объекты внутри кластеров похожи друг на друга и отличаются от объектов других кластеров.

Davies-Bouldin Index — показатель качества кластеризации, основанный на расстояниях между кластерами.

Adjusted Rand Index (ARI) — сравнение с эталонной разметкой.

4. Общие метрики и показатели:

Время отклика и производительность — важны для систем в реальном времени.

Надёжность и устойчивость — способность системы работать без сбоев.

Интерпретируемость — насколько легко понять и объяснить решения системы.

Пользовательская удовлетворённость — субъективная метрика, важная для прикладных систем.

1. Как обеспечить интерпретируемость решений нейронной сети?

1. Использование простых и прозрачных моделей

По возможности выбирайте более простые архитектуры (например, небольшие полносвязные сети), которые легче анализировать.

Иногда достаточно заменить нейросеть на более интерпретируемую модель (дерево решений, линейная регрессия), если точность приемлема.

2. Визуализация активаций и весов

Анализируйте активации нейронов и веса слоев, особенно в сверточных сетях — можно визуализировать, какие участки изображения влияют на решение (например, с помощью карт активаций, feature maps).

Используйте методы визуализации, такие как Grad-CAM, чтобы показать важные области входных данных.

3. Методы объяснения решений (Explainable AI, XAI)

LIME (Local Interpretable Model-agnostic Explanations): строит локальные линейные приближения модели для объяснения отдельных предсказаний.

SHAP (SHapley Additive exPlanations): оценивает вклад каждого признака в итоговое решение модели, основан на теории кооперативных игр.

Integrated Gradients: вычисляет интеграл градиентов модели по входным признакам для оценки их важности.

Anchors: генерирует правила (якоря), которые объясняют предсказания модели.

4. Использование внимания (Attention)

В моделях с механизмом внимания можно анализировать веса внимания, чтобы понять, на какие части входа модель «смотрит» при принятии решения.

5. Разбиение на подзадачи

Делите сложную задачу на более простые этапы, где решения каждого этапа проще интерпретировать.

6. Документирование и тестирование

Ведите документацию по архитектуре, параметрам и обучающим данным.

Проводите тесты с контролируемыми изменениями входных данных для проверки реакции модели.

7. Обучение с интерпретируемыми признаками

Используйте осмысленные, понятные человеку признаки, чтобы легче было анализировать влияние каждого из них.