Автономная некоммерческая организация высшего образования
«Волгоградский институт бизнеса»

Рабочая программа учебной дисциплины

Инструменты решения задач искусственного интеллекта

(Наименование дисциплины)

09.03.03 Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект»

(Направление подготовки / Профиль)

Бакалавр

(Квалификация)

Кафедра разработчик Экономики и управления

Год набора **2026**

Вид учебной деятельности	Трудоемкость (объем) дисциплины	
	Очная форма	Очно-заочная форма
	д	в
Зачетные единицы	**6**	**6**
Общее количество часов	216	216
Аудиторные часы контактной работы обучающегося с преподавателями:	64	36
– Лекционные (Л)	32	18
– Практические (ПЗ)	32	18
– Лабораторные (ЛЗ)		
– Семинарские (СЗ)		
Самостоятельная работа обучающихся (СРО)	98	144
К (Р-Г) Р (П) (+;-)		
Тестирование (+;-)		
ДКР (+;-)		
Зачет (+;-)	+	+
Зачет с оценкой (+;- (Кол-во часов))		
Экзамен (+;- (Кол-во часов))	+(54)	+(36)

+

Волгоград 2026

Содержание

Раздел 1. Организационно-методический раздел

1.1. Цели освоения дисциплины

Дисциплина **«Инструменты решения задач искусственного интеллекта»** входит **в часть, формируемую участниками образовательных отношений Б1.В.08** подготовки обучающихся по направлению **Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект».**

Целью дисциплины является формирование **компетенций** (*в соответствии с ФГОС ВО и требованиями к результатам освоения основной профессиональной образовательной программы высшего образования (ОПОП ВО)*):

ОПК-2. Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности;

Дескрипторы общепрофессиональных компетенций:

ОПК-2.1 – Способен использовать современные информационные технологии и программные средства при решении задач автоматизации предметной области, включая специализированные библиотеки и фреймворки для разработки систем искусственного интеллекта

ОПК-2.2 – Способен применять информационные технологии и программные средства отечественного производства на практике, в том числе отечественные платформы и библиотеки для разработки решений в области искусственного интеллекта

Перечисленные компетенции формируются в процессе достижения **индикаторов компетенций:**

Обобщенная трудовая функция/ трудовая функция	**Код и наименование дескриптора компетенций**	**Код и наименование индикатора достижения компетенций (из ПС)**
.	**ОПК-2.1** – Способен использовать современные информационные технологии и программные средства при решении задач автоматизации предметной области, включая специализированные библиотеки и фреймворки для разработки систем искусственного интеллекта **ОПК-2.2** – Способен применять информационные технологии и программные средства отечественного производства на практике, в том числе отечественные платформы и библиотеки для разработки решений в области искусственного интеллекта	Знает ИД-1 ОПК-2.1 Современные информационные технологии и программные средства, используемые при решении задач автоматизации предметной области, включая специализированные библиотеки и фреймворки для разработки систем искусственного интеллекта (без привязки к профессиональному стандарту) ИД-2 ОПК-2.2 Состав и функциональные возможности информационных технологий и программных средств отечественного производства, включая отечественные платформы и библиотеки для разработки решений в области искусственного интеллекта (без привязки к профессиональному стандарту) Умеет ИД-3 ОПК-2.1 Использовать современные информационные технологии и программные средства, включая специализированные библиотеки и фреймворки, для решения задач автоматизации предметной области и разработки систем искусственного интеллекта (без привязки к профессиональному стандарту) ИД-4 ОПК-2.2 Применять на практике ин-

		формационные технологии и программные средства отечественного производства, включая отечественные платформы и библиотеки для разработки решений в области искусственного интеллекта (без привязки к профессиональному стандарту) Имеет навыки ИД-5 ОПК-2.1 Владение навыками работы с современными информационными технологиями и программными средствами, включая специализированные библиотеки и фреймворки, для автоматизации предметной области и разработки систем искусственного интеллекта (без привязки к профессиональному стандарту) ИД-6 ОПК-2.2 Владение навыками практического применения информационных технологий и программных средств отечественного производства, включая отечественные платформы и библиотеки для разработки решений в области искусственного интеллекта (без привязки к профессиональному стандарту)

1.2. Место дисциплины в структуре ОПОП ВО направления подготовки «09.03.03 Прикладная информатика», направленность (профиль) «Прикладной искусственный интеллект»

№	Предшествующие дисциплины (дисциплины, изучаемые параллельно)	Последующие дисциплины
1	*2*	*3*
1	Компьютерное зрение	Выполнение и защита выпускной квалификационной работы
2	Компьютерная лингвистика	Проектирование систем с использованием технологий искусственного интеллекта
3	Инструменты решения задач искусственного интеллекта	Учебная практика (Научно-исследовательская работа (получение первичных навыков научно-исследовательской работы))
4	Основы систем искусственного интеллекта	Производственная практика (Технологическая (проектно-технологическая) практика)
5		Производственная практика (Эксплуатационная практика)
6		Производственная практика (Преддипломная практика)

Последовательность формирования компетенций в указанных дисциплинах может быть изменена в зависимости от формы и срока обучения, а также преподавания с использованием дистанционных технологий обучения.

1.3. Нормативная документация

Рабочая программа учебной дисциплины составлена на основе:

- Федерального государственного образовательного стандарта высшего образования по направлению подготовки **09.03.03 Прикладная информатика**;
- Учебного плана направления подготовки **09.03.03 Прикладная информатика, направленность (профиль) «Прикладной искусственный интеллект»** 2026 года набора;
- Образца рабочей программы учебной дисциплины (приказ № 113-О от 01.09.2021 г.).

Раздел 2. Тематический план

Очная форма обучения (полный срок)

№	Тема дисциплины	Трудоемкость				
		Всего	Аудиторные занятия		СРО	Код индикатора и дескриптора достижения компетенций
			Л	ПЗ (ЛЗ, СЗ)		
1	*2*	*3*	*4*	*5*	*6*	*7*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	10	2	2	6	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	10	2	2	6	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	10	2	2	6	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	10	2	2	6	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
5	Scikit-learn: унифицированный API для классического машинного обучения	10	2	2	6	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	10	2	2	6	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	10	2	2	6	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	10	2	2	6	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	10	2	2	6	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	10	2	2	6	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	10	2	2	6	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL	10	2	2	6	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	10	2	2	6	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe	10	2	2	6	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	10	2	2	6	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
16	Большие языковые модели и RAG-пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)	12	2	2	8	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2

Вид промежуточной аттестации (Зачет)	**+**				
Вид промежуточной аттестации (Экзамен)	**+(54)**				
Итого	**216**	**32**	**32**	**98**	

Очно-заочная форма обучения (полный срок)

№	Тема дисциплины	Трудоемкость				
		Всего	Аудиторные занятия		СРО	Код индикатора и дескриптора достижения компетенций
			Л	ПЗ (ЛЗ, СЗ)		
1	*2*	*3*	*4*	*5*	*6*	*7*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	10	2		8	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	10	2		8	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	10	2		8	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	10	2		8	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
5	Scikit-learn: унифицированный API для классического машинного обучения	10	2		8	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	10	2		8	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	10	2		8	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	12	2	2	8	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	14	2	2	10	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	12		2	10	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	12		2	10	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL	12		2	10	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	12		2	10	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe	12		2	10	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	12		2	10	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
16	Большие языковые модели и RAG-	12		2	10	ИД-5 ОПК-2.1

	пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)					ИД-6 ОПК-2.2
Вид промежуточной аттестации (Зачет)		+				
Вид промежуточной аттестации (Экзамен)		**+(36)**				
Итого		**216**	**18**	**18**	**144**	

Раздел 3. Содержание дисциплины

3.1. Содержание дисциплины

Тема 1. Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab. Обзор языков программирования, используемых в искусственном интеллекте. Python как основной язык: преимущества (простота синтаксиса, богатая экосистема библиотек, большое сообщество). R как язык для статистического анализа и визуализации. Julia как высокопроизводительный язык для научных вычислений. Сравнение языков по производительности, удобству и области применения. Среды разработки: Jupyter Notebook (интерактивная разработка, смешивание кода, визуализаций и текста), VS Code (профессиональная IDE с поддержкой отладки и Git), Google Colab (облачная среда с бесплатным GPU/TPU). Настройка окружения: менеджеры пакетов pip, conda, виртуальные окружения (venv, poetry). Установка ключевых библиотек. Обзор экосистемы: как взаимодействуют между собой библиотеки (NumPy -> Pandas -> Scikit-learn -> PyTorch). Ресурсы для изучения: документация, Kaggle, Stack Overflow.

Тема 2. NumPy и SciPy: фундамент численных вычислений и научных расчётов. Библиотека NumPy как основа для всех численных вычислений в Python. Многомерные массивы (ndarray): атрибуты (shape, dtype, size, ndim), способы создания (array, zeros, ones, arange, linspace, random). Индексирование и срезы, булево индексирование, fancy indexing. Векторизация вычислений и универсальные функции (ufunc). Правила трансляции (broadcasting). Операции линейной алгебры: dot, matmul, linalg.inv, linalg.det, linalg.eig. Библиотека SciPy как расширение NumPy для научных вычислений: модули оптимизации (optimize), интеграции (integrate), интерполяции (interpolate), статистики (stats), обработки сигналов (signal), разреженных матриц (sparse). Примеры: решение систем уравнений, оптимизация функций, статистические тесты.

Тема 3. Pandas и Polars: высокопроизводительная обработка и анализ табличных данных. Pandas как основная библиотека для работы с табличными и временными данными. Структуры Series (одномерный массив с метками) и DataFrame (двумерная таблица). Создание DataFrame из словарей, списков, CSV, Excel. Просмотр данных: head, tail, info, describe. Индексация и выборка: loc (по меткам), iloc (по позиции), булева маскировка. Работа с пропусками: isnull, dropna, fillna. Группировка и агрегация: groupby, agg, transform. Объединение данных: concat, merge (inner, left, right, outer). Создание сводных таблиц (pivot_table). Polars как высокопроизводительная альтернатива Pandas: ленивые и жадные вычисления, поддержка больших данных, работа с Apache Arrow. Сравнение производительности Pandas и Polars. Выбор инструмента в зависимости от объёма данных и требований к скорости.

Тема 4. Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly. Роль визуализации в анализе данных и интерпретации моделей. Matplotlib: фундаментальная библиотека для построения графиков. Фигура (figure) и оси (axes). Основные типы графиков: линейный (plot), точечный (scatter), столбчатый (bar), гистограмма (hist), ящик с усами (boxplot). Настройка внешнего вида: заголовки, подписи осей, легенда, сетка, цветовые карты. Создание нескольких подграфиков (subplots). Seaborn: высокоуровневый API для статистической визуализации. Улучшенные стили по умолчанию. Функции: pairplot (матрица парных графиков), heatmap (тепловая карта корреляций), violinplot, swarmplot. Plotly: интерактивная визуализация для веб. Построение графиков с возможностью масштабирования, панорамирования, отображения подсказок. Дашбор-

ды и визуализация в реальном времени.

Тема 5. Scikit-learn: унифицированный API для классического машинного обучения. Scikit-learn как стандартная библиотека для классических алгоритмов машинного обучения. Единый API: fit (обучение), predict (предсказание), transform (преобразование), fit_transform. Предобработка данных: масштабирование (StandardScaler, MinMaxScaler), кодирование категориальных признаков (OneHotEncoder, OrdinalEncoder). Разделение данных: train_test_split, кросс-валидация (cross_val_score). Основные алгоритмы: линейная и логистическая регрессия, деревья решений, случайный лес, градиентный бустинг, метод опорных векторов (SVM), k-ближайших соседей (KNN). Метрики качества: accuracy, precision, recall, f1_score (классификация); mean_squared_error, r2_score (регрессия). Подбор гиперпараметров: GridSearchCV, RandomizedSearchCV. Pipeline для объединения этапов обработки и моделирования.

Тема 6. XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных. Градиентный бустинг как один из самых эффективных методов для табличных данных. Основные принципы: последовательное обучение деревьев, каждое следующее дерево корректирует ошибки предыдущих. XGBoost (eXtreme Gradient Boosting): история, особенности (регуляризация, обработка пропусков, параллелизация, возможность использования GPU). Параметры: learning_rate, n_estimators, max_depth, subsample. LightGBM: оптимизация для больших данных, использование гистограмм, Growth strategy leaf-wise (в отличие от depth-wise), категориальные признаки без one-hot кодирования. CatBoost: автоматическая обработка категориальных признаков с помощью целевого кодирования с упорядочиванием (ordered target encoding), борьба с переобучением. Сравнение производительности и точности трёх реализаций. Практические рекомендации по выбору и настройке.

Тема 7. PyTorch: динамические вычислительные графы и исследовательские прототипы. PyTorch как фреймворк глубокого обучения, разработанный Facebook. Основные принципы: динамические вычислительные графы (define-by-run), удобство отладки, близость к Python. Тензоры (torch.Tensor): создание, операции, сходство с NumPy. Автоматическое дифференцирование (autograd): вычислительные графы, обратное распространение, вычисление градиентов. Построение нейронных сетей через torch.nn: модули (Linear, Conv2d, RNN), функции активации (ReLU, Sigmoid, Tanh), функции потерь (MSELoss, CrossEntropyLoss). Оптимизаторы: SGD, Adam, AdamW. Обучение модели: цикл по эпохам и батчам, forward pass, backward pass, обновление весов. DataLoader для загрузки данных. Сохранение и загрузка моделей (state_dict). Экосистема PyTorch: torchvision (модели и датасеты для CV), torchtext (NLP), torchaudio (аудио). Сравнение с TensorFlow: динамические vs статические графы, удобство для исследований.

Тема 8. TensorFlow и Keras: промышленное развёртывание и высокоуровневый API. TensorFlow как фреймворк глубокого обучения от Google. Эволюция: от статических графов к eager execution (динамическим вычислениям). Keras как высокоуровневый API (встроен в TensorFlow как tf.keras). Преимущества Keras: простота, модульность, прототипирование. Sequential API: последовательное добавление слоёв. Functional API: создание сложных нелинейных архитектур. Model Subclassing: полная гибкость для исследовательских задач. Обучение модели: compile (оптимизатор, функция потерь, метрики), fit (обучение), evaluate (оценка), predict (предсказание). Callbacks: EarlyStopping, ModelCheckpoint, TensorBoard. Загрузка предобученных моделей (MobileNet, ResNet) для transfer learning. TensorFlow Serving: развёртывание моделей в продакшене. TensorFlow Lite: модели для мобильных и встраиваемых устройств. Сравнение PyTorch и TensorFlow/Keras: исследовательские возможности vs промышленная надёжность.

Тема 9. Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка. Hugging Face как стандарт для работы с предобученными трансформерами. Библиотека transformers: единый API для загрузки, использования и донастройки моделей. Основные классы: AutoModel, AutoTokenizer, AutoConfig. Архитектуры: BERT (двунаправленный, для понимания текста), GPT (авторегрессивный, для генерации), T5 (энкодер-декодер), RoBERTa, DistilBERT,

ALBERT. Предобучение и тонкая настройка (fine-tuning). Pipeline API: высокоуровневый интерфейс для типовых задач: classification, token classification (NER), question-answering, text-generation, summarization, translation, sentiment analysis. Токенизаторы: WordPiece, Byte-Pair Encoding (BPE), SentencePiece. Работа с датасетами: библиотека datasets. Метрики: библиотека evaluate. Hub: репозиторий предобученных моделей и датасетов. Оптимизация инференса: квантизация, ONNX, использование GPU. Практика: донастройка BERT для задачи классификации текстов.

Тема 10. Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision. Обзор инструментов для обработки изображений. Pillow (PIL): базовая библиотека для загрузки, сохранения и простых манипуляций с изображениями (изменение размера, поворот, обрезка, фильтры). OpenCV: мощная библиотека для компьютерного зрения. Загрузка изображений и видео. Основные операции: преобразование цветовых пространств (RGB, BGR, HSV, Grayscale), геометрические преобразования, фильтрация (размытие, выделение границ), морфологические операции, обнаружение контуров. Детекция объектов с использованием предобученных каскадов Хаара (OpenCV). Torchvision: библиотека PyTorch для CV. Датасеты (ImageFolder, CIFAR10, MNIST). Трансформации (transforms.Compose): Resize, ToTensor, Normalize, RandomHorizontalFlip, ColorJitter. Предобученные модели: ResNet, VGG, EfficientNet, MobileNet. Визуализация карт активаций. Интеграция OpenCV и Torchvision для построения полного пайплайна обработки изображений.

Тема 11. Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim. Инструменты для работы с текстовыми данными. NLTK (Natural Language Toolkit): академическая библиотека для обучения NLP. Корпуса текстов, токенизация, стемминг (Porter), лемматизация (WordNet), POS-тегирование, выделение именованных сущностей (NER). spaCy: промышленная библиотека для быстрой обработки текста. Модели для многих языков (включая русский). Конвейер обработки: токенизация, лемматизация, POS-тегирование, синтаксический анализ (зависимости), NER. Визуализация с помощью displacy. Сравнение spaCy и NLTK. Gensim: библиотека для тематического моделирования и векторных представлений слов. Реализация Word2Vec (CBOW, Skip-gram), FastText, Doc2Vec. LDA (Latent Dirichlet Allocation) для тематического моделирования. Применение: поиск семантических близостей, кластеризация текстов.

Тема 12. Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL. Представление графовых данных: вершины, рёбра, атрибуты. NetworkX: библиотека для создания, анализа и визуализации графов. Основные алгоритмы: поиск кратчайших путей, компоненты связности, центральности, кластеризация. Графовые нейронные сети (GNN): применение для прогнозирования свойств узлов, рёбер и целых графов. PyTorch Geometric: библиотека для глубокого обучения на графах на основе PyTorch. Основные классы: Data (граф), DataLoader. Конволюционные слои: GCN (Graph Convolutional Network), GAT (Graph Attention Network), GraphSAGE. Задачи: классификация узлов (Cora, CiteSeer), классификация графов (MUTAG, PROTEINS). DGL (Deep Graph Library): альтернативная библиотека от Amazon, поддерживает PyTorch, TensorFlow, MXNet. Сравнение PyTorch Geometric и DGL. Применение GNN: социальные сети, молекулярная химия, рекомендательные системы, анализ транспортных сетей.

Тема 13. MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna. Проблема воспроизводимости экспериментов в машинном обучении. MLflow: платформа для управления жизненным циклом ML. Компоненты: MLflow Tracking (логирование параметров, метрик, артефактов), MLflow Projects (упаковка кода), MLflow Models (управление моделями), MLflow Registry (реестр моделей). DVC (Data Version Control): контроль версий данных и пайплайнов. Хранение больших файлов в облаке (S3, GCS) и метаданных в Git. Воспроизводимость экспериментов. Weights & Biases (wandb): облачная платформа для логирования и визуализации экспериментов. Визуализация кривых обучения, гиперпараметров, сравнение экспериментов. Optuna: фреймворк для автоматической оптимизации гиперпараметров. Алгоритмы: случайный поиск, TPE (Tree-structured Parzen Estimator), CMA-ES, ранняя остановка (pruning). Интеграция с MLflow и wandb. Сравнение инструментов и выбор в зависимости от масштаба команды и проекта.

Тема 14. Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe. Способы развёртывания моделей машинного обучения в продакшене. FastAPI: современный веб-фреймворк для создания REST API. Асинхронность, автоматическая документация (Swagger), валидация данных с Pydantic. Реализация эндпоинта /predict, загрузка модели при старте сервера. ONNX (Open Neural Network Exchange): формат для обмена моделями между фреймворками. Конвертация моделей из PyTorch и TensorFlow в ONNX. Инференс с ONNX Runtime (ускорение на CPU/GPU). Контейнеризация с Docker: Dockerfile, образы, зависимости. Оркестрация с Kubernetes: развёртывание, масштабирование, балансировка нагрузки. TensorFlow Serving: высокопроизводительный сервер для моделей TensorFlow. Поддержка версионирования моделей, gRPC/REST API. TorchServe: официальный сервер инференса для PyTorch. Упаковка моделей с помощью torch-model-archiver. Сравнение подходов: когда использовать FastAPI, а когда специализированные серверы.

Тема 15. Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI. Обзор облачных платформ и сервисов для машинного обучения. Kaggle: соревнования по DS, датасеты, ноутбуки с бесплатным GPU/TPU. Hugging Face Spaces: хостинг для демонстрации ML-приложений. Интеграция с Gradio и Streamlit. Возможность развёртывания моделей Transformer. Yandex DataSphere: российская платформа для end-to-end ML. Интегрированная среда разработки, управление вычислительными ресурсами (CPU/GPU), пайплайны данных, интеграция с Yandex Cloud. Google Vertex AI: платформа от Google для всего жизненного цикла ML: подготовка данных, обучение, настройка гиперпараметров, развёртывание, мониторинг. AutoML для автоматического построения моделей. Сравнение платформ по функциональности, стоимости, удобству использования.

Тема 16. Большие языковые модели и RAG-пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant). Проблемы использования больших языковых моделей (LLM): ограниченный контекст, галлюцинации, отсутствие актуальных знаний. RAG (Retrieval-Augmented Generation): архитектура для повышения качества ответов через поиск релевантного контекста. Компоненты RAG: эмбеддинг-модель, векторная база данных, ретривер, LLM-генератор. LangChain: фреймворк для создания приложений на основе LLM. Цепочки (chains), агенты, инструменты, интеграция с векторными БД. LlamaIndex (GPT Index): специализированная библиотека для индексации и поиска по пользовательским документам. Векторные базы данных: Chroma (лёгкая, встроенная), FAISS (эффективная библиотека от Facebook), Qdrant (масштабируемая, клиент-серверная). Эмбеддинг-модели: text-embedding-ada-002 (OpenAI), Sentence-Transformers (sentence-transformers/all-MiniLM-L6-v2). Практика: построение RAG-пайплайна для вопросно-ответной системы по документам. Оценка качества RAG: метрики релевантности, точности ответов.

3.2. Содержание практического блока дисциплины

Очная форма обучения (полный срок)

№	Тема практического (семинарского, лабораторного) занятия
1	*2*
ПЗ 1	Практическое занятие: Установка Python через Anaconda и создание виртуального окружения с помощью venv и conda. Установка Jupyter Notebook и VS Code с расширениями Python, Pylance, Jupyter. Знакомство с интерфейсом Google Colab: создание нового ноутбука, подключение GPU/TPU, установка библиотек через !pip. Написание простого скрипта на Python (вычисление факториала, работа со списками). Сравнение выполнения аналогичной задачи на R и Julia (через базовые инсталляции или онлайн-среды). Создание requirements.txt и установка зависимостей. Практика работы с Git: инициализация репозитория, commit, push в GitHub. Настройка .gitignore для

	Python-проектов.
ПЗ 2	Практическое занятие: Создание массивов NumPy с помощью array, zeros, ones, arange, linspace, random.randn. Изучение атрибутов shape, dtype, size, ndim. Выполнение индексирования, срезов, булевой фильтрации и fancy indexing. Векторизованные вычисления: замена цикла на операции с массивами, сравнение скорости выполнения. Применение универсальных функций (sqrt, exp, sin, log). Решение задач на трансляцию: сложение массива и числа, матрицы и вектора. Вычисление матричных операций: умножение (dot), определитель (linalg.det), обратная матрица (linalg.inv), собственные числа (linalg.eig). Использование SciPy: решение системы линейных уравнений (linalg.solve), минимизация функции (optimize.minimize), интерполяция (interpolate.interp1d), статистические тесты (stats.ttest_ind). Сравнение производительности SciPy и чисто Python реализации.
ПЗ 3	Практическое занятие: Загрузка датасетов из CSV (read_csv), Excel (read_excel), JSON. Первичный анализ данных: head, tail, info, describe. Индексация и выборка с loc и iloc. Фильтрация строк по условиям (булевы маски). Обработка пропусков: обнаружение с isnull, удаление dropna, заполнение fillna (средним, медианой, методом ffill/bfill). Группировка и агрегация: groupby с mean, sum, count, применение agg для нескольких функций. Объединение данных: concat (вертикальное/горизонтальное), merge (inner, left, right, outer). Создание сводных таблиц (pivot_table). Знакомство с Polars: загрузка данных, выполнение аналогичных операций, сравнение скорости выполнения на большом датасете (например, 1 млн строк). Визуализация результатов группировки с помощью plot.
ПЗ 4	Практическое занятие: Построение линейного графика (plot) и точечной диаграммы (scatter) с Matplotlib. Настройка заголовков, подписей осей, легенды, сетки, цветов. Создание столбчатой диаграммы (bar) и гистограммы (hist). Построение ящика с усами (boxplot) для выявления выбросов. Создание нескольких подграфиков на одной фигуре (subplots). Использование Seaborn: установка стиля (set_style), построение матрицы парных графиков (pairplot), тепловой карты корреляций (heatmap), скрипичного графика (violinplot). Построение интерактивных графиков с Plotly: scatter, line, bar, добавление подсказок при наведении. Создание дашборда с несколькими связанными графиками. Сохранение графиков в файл (png, html). Анализ датасета по выбору с полным циклом визуализации.
ПЗ 5	Практическое занятие: Загрузка встроенных датасетов (load_iris, load_digits, fetch_california_housing). Разделение данных на обучающую и тестовую выборки (train_test_split). Масштабирование признаков (StandardScaler, MinMaxScaler) и кодирование категориальных переменных (OneHotEncoder). Обучение моделей: линейная регрессия (LinearRegression), логистическая регрессия (LogisticRegression), дерево решений (DecisionTreeClassifier), случайный лес (RandomForestClassifier). Оценка качества с помощью метрик: accuracy_score, precision_score, recall_score, f1_score, mean_squared_error, r2_score. Кросс-валидация (cross_val_score). Подбор гиперпараметров с GridSearchCV и RandomizedSearchCV. Создание Pipeline для объединения масштабирования и модели. Сохранение и загрузка модели с joblib. Сравнение нескольких моделей на одном датасете и выбор лучшей.
ПЗ 6	Практическое занятие: Установка XGBoost, LightGBM, CatBoost. Загрузка структурированного датасета (например, Titanic, Credit Card Fraud). Обучение модели XGBoost с параметрами по умолчанию, оценка качества. Настройка гиперпараметров: learning_rate, n_estimators, max_depth, subsample, colsample_bytree. Визуализация важности признаков (feature_importances_). Обучение LightGBM: указание categorical_feature для категориальных столбцов, использование параметра leaf-wise. Обучение CatBoost: автоматическая обработка категориальных признаков (cat_features). Сравнение времени обучения и качества (AUC-ROC, accuracy, F1) трёх моделей на одном датасете. Использование ранней остановки (early_stopping_rounds) и валидационного набора. Подбор гиперпараметров с Optuna для одной из моделей. Сохранение лучшей модели.

ПЗ 7	Практическое занятие: Установка PyTorch (CPU/GPU версия). Создание тензоров и операции над ними (сходство с NumPy). Использование autograd: вычисление градиентов для простой функции (y = x^2). Построение простой нейронной сети с одним скрытым слоем с использованием torch.nn.Module. Выбор функции потерь (CrossEntropyLoss для классификации) и оптимизатора (Adam). Обучение модели на MNIST: цикл по эпохам и батчам, forward pass, backward pass, обновление весов. Отслеживание функции потерь и точности на валидации. Использование DataLoader для загрузки батчей. Визуализация кривых обучения. Сохранение и загрузка модели (state_dict). Загрузка предобученной модели ResNet из torchvision и fine-tuning на пользовательском датасете (например, кошки против собак). Сравнение обучения с нуля и transfer learning.
ПЗ 8	Практическое занятие: Установка TensorFlow. Построение Sequential модели: добавление слоев Dense, Dropout, BatchNormalization, функций активации. Компиляция модели: выбор оптимизатора (Adam), функции потерь, метрик (accuracy). Обучение модели (fit) с указанием эпох, валидационных данных. Использование callbacks: EarlyStopping (остановка при отсутствии улучшений), ModelCheckpoint (сохранение лучшей модели), TensorBoard (логирование). Functional API: создание модели с несколькими входами и выходами (например, модель для изображения и текста). Model Subclassing: полный контроль над forward pass. Загрузка предобученной модели MobileNetV2 из tf.keras.applications, заморозка базовых слоёв, добавление новых слоёв для fine-tuning. Сохранение модели в формате SavedModel и конвертация в TensorFlow Lite. Развёртывание модели с TensorFlow Serving (локальный сервер). Сравнение Keras и PyTorch на примере одинаковой архитектуры.
ПЗ 9	Практическое занятие: Установка библиотек transformers, datasets, evaluate, tokenizers, accelerate. Загрузка предобученной модели BERT и токенизатора (bert-base-uncased). Применение pipeline для задач: sentiment-analysis, ner (выделение именованных сущностей), question-answering, text-generation. Тонкая настройка BERT для задачи классификации текстов (например, IMDB reviews). Подготовка датасета: токенизация с truncation и padding. Использование Trainer API или стандартного цикла обучения PyTorch. Оценка качества с помощью accuracy и F1. Загрузка GPT-2 (gpt2-medium) для генерации текста: использование параметров max_length, temperature, top_k, top_p. Fine-tuning GPT-2 на собственном корпусе текстов. Использование модели T5 для суммаризации текста. Работа с русскоязычными моделями: ruBERT, ruGPT-3 (или аналогичными). Экспорт модели в ONNX для ускорения инференса.
ПЗ 10	Практическое занятие: Загрузка, сохранение и отображение изображений с Pillow и OpenCV. Преобразование цветовых пространств (BGR ↔ RGB, в градации серого). Геометрические преобразования: resize, rotate, crop, translation. Применение фильтров: размытие (GaussianBlur), выделение границ (Canny). Морфологические операции: эрозия, дилатация, открытие, закрытие. Обнаружение контуров (findContours) и их отрисовка. Детекция лиц с использованием каскадов Хаара (haarcascade_frontalface_default). Torchvision: загрузка датасета CIFAR10 с torchvision.datasets. Применение трансформаций: Compose(Resize, RandomHorizontalFlip, ToTensor, Normalize). Загрузка предобученной модели ResNet-18, замена последнего слоя под количество классов. Fine-tuning на пользовательском датасете (например, классификация видов цветов). Визуализация карт активации сверточных слоёв. Интеграция OpenCV для предобработки и Torchvision для модели в едином пайплайне.
ПЗ 11	Практическое занятие: Установка NLTK и загрузка корпусов (punkt, stopwords, wordnet, averaged_perceptron_tagger). Токенизация слов и предложений. Удаление стоп-слов. Стемминг (PorterStemmer) и лемматизация (WordNetLemmatizer). POS-тегирование частей речи. Выделение именованных сущностей (ne_chunk). Установка spaCy и загрузка модели (en_core_web_sm, ru_core_news_sm). Конвейер обработки: токенизация, лемматизация, POS-тегирование, синтаксический анализ (зависимости), NER. Визуализация зависимостей и NER с displacy. Сравнение скорости обработки

	spaCy и NLTK на большом тексте. Gensim: обучение Word2Vec на корпусе текстов. Визуализация ближайших слов к заданному (most_similar). Обучение LDA для тематического моделирования, визуализация тем. Применение Doc2Vec для векторизации документов.
ПЗ 12	Практическое занятие: Создание графа с NetworkX: добавление вершин и рёбер, задание атрибутов. Визуализация графа с различными layout (spring, circular, random). Вычисление характеристик: степень вершин, центральность (betweenness, closeness), поиск кратчайшего пути (shortest_path). Алгоритмы: поиск компонент связности, кластеризация, поиск клик. PyTorch Geometric: установка. Создание объекта Data из списка вершин, рёбер и атрибутов. Загрузка встроенного датасета (Cora, CiteSeer). Реализация GCN (Graph Convolutional Network) для задачи классификации узлов. Обучение модели и оценка точности. Визуализация эмбеддингов узлов после обучения (с помощью t-SNE). DGL: установка и аналогичный пример классификации узлов. Сравнение API PyTorch Geometric и DGL. Реализация GAT (Graph Attention Network) для задачи классификации графов (MUTAG). Применение GNN для предсказания свойств молекул (например, растворимости).
ПЗ 13	Практическое занятие: Установка MLflow. Логирование эксперимента: параметры (params), метрики (metrics), артефакты (модель, графики). Запуск MLflow UI для визуализации результатов. Сравнение нескольких экспериментов. DVC: инициализация (dvc init), добавление данных (dvc add), сохранение в удалённое хранилище (S3, GCS). Создание пайплайна (dvc run). Воспроизведение эксперимента (dvc repro). Weights & Biases: установка wandb, логирование экспериментов через wandb.init, wandb.log. Визуализация кривых обучения в реальном времени. Сравнение запусков. Optuna: определение функции цели (objective) с предложением гиперпараметров (suggest_int, suggest_float). Оптимизация с созданием исследования (study.optimize). Визуализация результатов оптимизации (plot_parallel_coordinate, plot_contour). Интеграция Optuna с MLflow для автоматического логирования. Полный пайплайн: Optuna подбирает гиперпараметры, MLflow логирует результаты, DVC управляет данными.
ПЗ 14	Практическое занятие: Создание REST API на FastAPI: эндпоинт /predict, загрузка модели при старте, валидация входных данных с Pydantic. Тестирование API с помощью Swagger UI (/docs) и Postman. Конвертация модели PyTorch в ONNX: torch.onnx.export. Инференс с ONNX Runtime: сравнение скорости с PyTorch. Создание Dockerfile для FastAPI приложения: выбор базового образа (python:3.10-slim), копирование зависимостей, команда запуска. Сборка образа (docker build) и запуск контейнера (docker run). Публикация образа на Docker Hub. TensorFlow Serving: сохранение модели в формате SavedModel, запуск сервера через Docker, отправка gRPC/REST запросов. TorchServe: упаковка модели PyTorch (torch-model-archiver), запуск сервера, тестирование предсказаний. Сравнение простоты настройки и производительности FastAPI, TensorFlow Serving и TorchServe.
ПЗ 15	Практическое занятие: Регистрация на Kaggle, участие в соревновании (Titanic). Создание и выполнение ноутбука с GPU. Загрузка датасетов с Kaggle. Hugging Face Spaces: создание нового Space, выбор SDK (Gradio, Streamlit). Развёртывание простого приложения для классификации изображений с Gradio. Интеграция модели из Hugging Face Hub. Yandex DataSphere: создание проекта, настройка вычислительных ресурсов (CPU/GPU). Импорт данных из S3, обучение модели в ноутбуке, сохранение артефактов. Настройка пайплайна для регулярного переобучения. Google Vertex AI: загрузка данных в Cloud Storage, создание Dataset, обучение модели с AutoML или пользовательским кодом. Развёртывание модели в эндпоинт, отправка тестовых запросов. Сравнение стоимости и функциональности платформ.
ПЗ 16	Практическое занятие: Установка LangChain. Создание цепочки (chain) для генерации текста с использованием OpenAI GPT (или локальной модели). Интеграция с поиском в интернете через инструменты (Tools). Установка Chroma: создание векторной базы данных из документов (txt, pdf). Создание эмбеддингов с

	SentenceTransformers и добавление в Chroma. Построение RAG-пайплайна: ретривер из Chroma + LLM для генерации ответа. LangChain Expression Language (LCEL) для создания цепочек. LlamaIndex: индексация документов, создание query engine. Сравнение результатов запросов с RAG и без RAG. FAISS: построение индекса для миллиона векторов, поиск ближайших соседей. Qdrant: запуск через Docker, создание коллекции, добавление векторов, поиск. Оценка качества RAG: метрики релевантности (hit rate, MRR). Развёртывание RAG-приложения с FastAPI и Streamlit. Обсуждение ограничений: размер контекста, задержки, качество эмбеддингов.

Очно-заочная форма обучения (полный срок)

№	Тема практического (семинарского, лабораторного) занятия
1	***2***
ПЗ 8	Практическое занятие: Установка TensorFlow. Построение Sequential модели: добавление слоев Dense, Dropout, BatchNormalization, функций активации. Компиляция модели: выбор оптимизатора (Adam), функции потерь, метрик (accuracy). Обучение модели (fit) с указанием эпох, валидационных данных. Использование callbacks: EarlyStopping (остановка при отсутствии улучшений), ModelCheckpoint (сохранение лучшей модели), TensorBoard (логирование). Functional API: создание модели с несколькими входами и выходами (например, модель для изображения и текста). Model Subclassing: полный контроль над forward pass. Загрузка предобученной модели MobileNetV2 из tf.keras.applications, заморозка базовых слоёв, добавление новых слоёв для fine-tuning. Сохранение модели в формате SavedModel и конвертация в TensorFlow Lite. Развёртывание модели с TensorFlow Serving (локальный сервер). Сравнение Keras и PyTorch на примере одинаковой архитектуры.
ПЗ 9	Практическое занятие: Установка библиотек transformers, datasets, evaluate, tokenizers, accelerate. Загрузка предобученной модели BERT и токенизатора (bert-base-uncased). Применение pipeline для задач: sentiment-analysis, ner (выделение именованных сущностей), question-answering, text-generation. Тонкая настройка BERT для задачи классификации текстов (например, IMDB reviews). Подготовка датасета: токенизация с truncation и padding. Использование Trainer API или стандартного цикла обучения PyTorch. Оценка качества с помощью accuracy и F1. Загрузка GPT-2 (gpt2-medium) для генерации текста: использование параметров max_length, temperature, top_k, top_p. Fine-tuning GPT-2 на собственном корпусе текстов. Использование модели T5 для суммаризации текста. Работа с русскоязычными моделями: ruBERT, ruGPT-3 (или аналогичными). Экспорт модели в ONNX для ускорения инференса.
ПЗ 10	Практическое занятие: Загрузка, сохранение и отображение изображений с Pillow и OpenCV. Преобразование цветовых пространств (BGR ↔ RGB, в градации серого). Геометрические преобразования: resize, rotate, crop, translation. Применение фильтров: размытие (GaussianBlur), выделение границ (Canny). Морфологические операции: эрозия, дилатация, открытие, закрытие. Обнаружение контуров (findContours) и их отрисовка. Детекция лиц с использованием каскадов Хаара (haarcascade_frontalface_default). Torchvision: загрузка датасета CIFAR10 с torchvision.datasets. Применение трансформаций: Compose(Resize, RandomHorizontalFlip, ToTensor, Normalize). Загрузка предобученной модели ResNet-18, замена последнего слоя под количество классов. Fine-tuning на пользовательском датасете (например, классификация видов цветов). Визуализация карт активации сверточных слоёв. Интеграция OpenCV для предобработки и Torchvision для модели в едином пайплайне.
ПЗ 11	Практическое занятие: Установка NLTK и загрузка корпусов (punkt, stopwords, wordnet, averaged_perceptron_tagger). Токенизация слов и предложений. Удаление стоп-слов. Стемминг (PorterStemmer) и лемматизация (WordNetLemmatizer). POS-тегирование частей речи. Выделение именованных сущностей (ne_chunk). Установка spaCy и загрузка модели (en_core_web_sm, ru_core_news_sm). Конвейер обработки: токенизация, лемматизация, POS-тегирование, синтаксический анализ (зависимости),

	NER. Визуализация зависимостей и NER с displacy. Сравнение скорости обработки spaCy и NLTK на большом тексте. Gensim: обучение Word2Vec на корпусе текстов. Визуализация ближайших слов к заданному (most_similar). Обучение LDA для тематического моделирования, визуализация тем. Применение Doc2Vec для векторизации документов.
ПЗ 12	Практическое занятие: Создание графа с NetworkX: добавление вершин и рёбер, задание атрибутов. Визуализация графа с различными layout (spring, circular, random). Вычисление характеристик: степень вершин, центральность (betweenness, closeness), поиск кратчайшего пути (shortest_path). Алгоритмы: поиск компонент связности, кластеризация, поиск клик. PyTorch Geometric: установка. Создание объекта Data из списка вершин, рёбер и атрибутов. Загрузка встроенного датасета (Cora, CiteSeer). Реализация GCN (Graph Convolutional Network) для задачи классификации узлов. Обучение модели и оценка точности. Визуализация эмбеддингов узлов после обучения (с помощью t-SNE). DGL: установка и аналогичный пример классификации узлов. Сравнение API PyTorch Geometric и DGL. Реализация GAT (Graph Attention Network) для задачи классификации графов (MUTAG). Применение GNN для предсказания свойств молекул (например, растворимости).
ПЗ 13	Практическое занятие: Установка MLflow. Логирование эксперимента: параметры (params), метрики (metrics), артефакты (модель, графики). Запуск MLflow UI для визуализации результатов. Сравнение нескольких экспериментов. DVC: инициализация (dvc init), добавление данных (dvc add), сохранение в удалённое хранилище (S3, GCS). Создание пайплайна (dvc run). Воспроизведение эксперимента (dvc repro). Weights & Biases: установка wandb, логирование экспериментов через wandb.init, wandb.log. Визуализация кривых обучения в реальном времени. Сравнение запусков. Optuna: определение функции цели (objective) с предложением гиперпараметров (suggest_int, suggest_float). Оптимизация с созданием исследования (study.optimize). Визуализация результатов оптимизации (plot_parallel_coordinate, plot_contour). Интеграция Optuna с MLflow для автоматического логирования. Полный пайплайн: Optuna подбирает гиперпараметры, MLflow логирует результаты, DVC управляет данными.
ПЗ 14	Практическое занятие: Создание REST API на FastAPI: эндпоинт /predict, загрузка модели при старте, валидация входных данных с Pydantic. Тестирование API с помощью Swagger UI (/docs) и Postman. Конвертация модели PyTorch в ONNX: torch.onnx.export. Инференс с ONNX Runtime: сравнение скорости с PyTorch. Создание Dockerfile для FastAPI приложения: выбор базового образа (python:3.10-slim), копирование зависимостей, команда запуска. Сборка образа (docker build) и запуск контейнера (docker run). Публикация образа на Docker Hub. TensorFlow Serving: сохранение модели в формате SavedModel, запуск сервера через Docker, отправка gRPC/REST запросов. TorchServe: упаковка модели PyTorch (torch-model-archiver), запуск сервера, тестирование предсказаний. Сравнение простоты настройки и производительности FastAPI, TensorFlow Serving и TorchServe.
ПЗ 15	Практическое занятие: Регистрация на Kaggle, участие в соревновании (Titanic). Создание и выполнение ноутбука с GPU. Загрузка датасетов с Kaggle. Hugging Face Spaces: создание нового Space, выбор SDK (Gradio, Streamlit). Развёртывание простого приложения для классификации изображений с Gradio. Интеграция модели из Hugging Face Hub. Yandex DataSphere: создание проекта, настройка вычислительных ресурсов (CPU/GPU). Импорт данных из S3, обучение модели в ноутбуке, сохранение артефактов. Настройка пайплайна для регулярного переобучения. Google Vertex AI: загрузка данных в Cloud Storage, создание Dataset, обучение модели с AutoML или пользовательским кодом. Развёртывание модели в эндпоинт, отправка тестовых запросов. Сравнение стоимости и функциональности платформ.

3.3. Образовательные технологии

Очная форма обучения (полный срок)

№	Тема занятия	Вид учебного занятия	Форма / Методы интерактивного обучения	% учебного времени
1	*2*	*3*	*4*	*5*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	ПЗ	Дискуссионные технологии, Работа в малых группах, Взаимопроверка, Мозговой штурм, Интерактивные тренажёры	25
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	ПЗ	Дискуссионные технологии, Групповое решение проблемных задач, Кейс-стади, Взаимообучение, Проектно-ориентированное обучение	25
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	ПЗ	Работа в парах, Деловая игра, Дискуссионные технологии, Мозговой штурм, Интерактивная визуализация с коллективным обсуждением	25
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	ПЗ	Семинар-дискуссия, Работа в группах с презентацией, Ролевая игра, Интерактивная доска, Кейс-стади	25
5	Scikit-learn: унифицированный API для классического машинного обучения	ПЗ	Групповое решение задач, Конкурс, Интерактивная визуализация, Мозговой штурм, Проектно-ориентированное обучение	25
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	ПЗ	Дискуссионные технологии, Работа в группах, Кейс-стади, Проектно-ориентированное обучение, Взаимооценка	25
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	ПЗ	Проектно-ориентированное обучение	25
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	ПЗ	Дискуссионные технологии, Работа в малых группах, Взаимопроверка, Мозговой штурм, Интерактивные тренажёры	25
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	ПЗ	Дискуссионные технологии, Групповое решение проблемных задач, Кейс-стади, Взаимообучение, Проектно-ориентированное обучение	25
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	ПЗ	Работа в парах, Деловая игра, Дискуссионные технологии, Мозговой штурм, Интерактивная визуализация с коллективным обсуждением	25
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	ПЗ	Семинар-дискуссия, Работа в группах с презентацией, Ролевая игра, Интерактивная доска, Кейс-стади	25
12	Инструменты для работы с	ПЗ	Групповое решение задач, Кон-	25

	графовыми данными: NetworkX, PyTorch Geometric, DGL		курс, Интерактивная визуализация, Мозговой штурм, Проектно-ориентированное обучение	
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	ПЗ	Дискуссионные технологии, Работа в группах, Кейс-стади, Проектно-ориентированное обучение, Взаимооценка	25
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe	ПЗ	Проектно-ориентированное обучение	25
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	ПЗ	Дискуссионные технологии, Работа в группах, Кейс-стади, Проектно-ориентированное обучение, Взаимооценка	25
16	Большие языковые модели и RAG-пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)	ПЗ	Проектно-ориентированное обучение	25
Итого				**25%**

Очно-заочная форма обучения (полный срок)

№	Тема занятия	Вид учебного занятия	Форма / Методы интерактивного обучения	% учебного времени
1	*2*	*3*	*4*	*5*
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	ПЗ	Дискуссионные технологии, Работа в малых группах, Взаимопроверка, Мозговой штурм, Интерактивные тренажёры	25
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	ПЗ	Дискуссионные технологии, Групповое решение проблемных задач, Кейс-стади, Взаимообучение, Проектно-ориентированное обучение	25
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	ПЗ	Работа в парах, Деловая игра, Дискуссионные технологии, Мозговой штурм, Интерактивная визуализация с коллективным обсуждением	25
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	ПЗ	Семинар-дискуссия, Работа в группах с презентацией, Ролевая игра, Интерактивная доска, Кейс-стади	25
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL	ПЗ	Групповое решение задач, Конкурс, Интерактивная визуализация, Мозговой штурм, Проектно-ориентированное обучение	25
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	ПЗ	Дискуссионные технологии, Работа в группах, Кейс-стади, Проектно-ориентированное обучение, Взаимооценка	25
14	Развёртывание моделей:	ПЗ	Проектно-ориентированное обу-	25

	FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe		чение	
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	ПЗ	Дискуссионные технологии, Работа в группах, Кейс-стади, Проектно-ориентированное обучение, Взаимооценка	25
Итого				**25%**

Раздел 4. Организация самостоятельной работы обучающихся

4.1. Организация самостоятельной работы обучающихся

№	Тема дисциплины	№ вопросов	№ рекомендуемой литературы
1	*2*	*3*	*4*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	1-5	1, 3, 6, 11, 14
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	6-10	1, 4, 6, 9, 11
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	11-15	1, 4, 6, 9, 11
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	16-20	3, 4, 6, 8, 10, 11, 12
5	Scikit-learn: унифицированный API для классического машинного обучения	21-25	3, 4, 6, 8, 10, 11, 12
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	26-30	2, 3, 4, 6, 10, 11, 13
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	31-35	2, 4, 11, 12
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	35-40	2, 3, 4, 7, 10, 11, 12, 13
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	41-45	2, 3, 4, 7, 10, 11, 12, 13
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	46-50	2, 3, 4, 6, 10, 11, 13
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	51-55	1, 3, 4, 6, 9, 11
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL	56-60	3, 4, 6, 11, 15
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	61-65	3, 4, 6, 11, 14, 15
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe	66-70	3, 4, 6, 11, 14, 15
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	71-75	2, 3, 4, 6, 10, 11, 13
16	Большие языковые модели и RAG-пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)	76-80	2, 3, 4, 6, 10, 11, 13

Перечень вопросов, выносимых на самостоятельную работу обучающихся

1. Какие языки программирования наиболее популярны в области искусственного интеллекта и почему Python является основным?
2. В чём преимущества использования Google Colab перед локальной средой разработки?
3. Что такое виртуальное окружение в Python и зачем оно нужно?
4. Какие основные отличия между Jupyter Notebook и VS Code как средами разработки для ИИ?
5. Как установить библиотеку через pip и как создать requirements.txt для проекта?
6. Что такое массив ndarray в NumPy и чем он отличается от списка Python?
7. Какие способы создания массивов NumPy вы знаете? Приведите примеры.

8. Что такое векторизация вычислений и каковы её преимущества перед циклами?
9. Сформулируйте правила трансляции (broadcasting) в NumPy.
10. Какие задачи можно решать с помощью модулей SciPy (optimize, integrate, stats, sparse)?
11. Что такое Series и DataFrame в Pandas? В чём их отличие?
12. Как выполнить фильтрацию строк DataFrame по условию с использованием булевой маски?
13. Какие методы Pandas используются для обнаружения и заполнения пропусков?
14. В чём разница между методами merge и concat в Pandas?
15. Каковы преимущества Polars перед Pandas для работы с большими объёмами данных?
16. Какие типы графиков можно построить с помощью Matplotlib и для каких задач?
17. В чём преимущества Seaborn перед Matplotlib для статистической визуализации?
18. Как создать интерактивный график с Plotly и в чём его преимущества?
19. Как построить несколько подграфиков в одной фигуре с помощью Matplotlib?
20. Что такое тепловая карта (heatmap) и для визуализации каких данных она применяется?
21. Как устроен единый API Scikit-learn (fit, predict, transform)?
22. Какие метрики используются для оценки качества классификации и регрессии?
23. Что такое кросс-валидация и зачем она нужна?
24. Как выполнить подбор гиперпараметров с помощью GridSearchCV?
25. Что такое Pipeline в Scikit-learn и какие преимущества он даёт?
26. В чём основное отличие градиентного бустинга от случайного леса?
27. Какие ключевые параметры влияют на качество модели XGBoost?
28. В чём преимущества LightGBM и CatBoost перед XGBoost?
29. Как CatBoost автоматически обрабатывает категориальные признаки?
30. Что такое ранняя остановка (early stopping) и как она используется при обучении бустингов?
31. В чём основное отличие PyTorch от TensorFlow?
32. Что такое динамический вычислительный граф и какие преимущества он даёт?
33. Как работает автоматическое дифференцирование (autograd) в PyTorch?
34. Что такое тензор в PyTorch и как он связан с NumPy?
35. Как сохранить и загрузить обученную модель PyTorch?
36. Какие типы API предоставляет Keras (Sequential, Functional, Subclassing)?
37. Что такое transfer learning и как его реализовать в Keras?
38. Для чего используются callbacks EarlyStopping и ModelCheckpoint?
39. Каковы преимущества TensorFlow Serving для промышленного развёртывания?
40. В чём отличие TensorFlow Lite от полноценного TensorFlow?
41. Что такое pipeline в библиотеке Transformers и для каких задач он используется?
42. Как выполнить тонкую настройку (fine-tuning) BERT для задачи классификации текстов?
43. Какие параметры влияют на качество генерации текста в GPT (temperature, top_k, top_p)?
44. Что такое токенизатор в контексте трансформеров и какие типы токенизаторов существуют?
45. Как загрузить предобученную русскоязычную модель из Hugging Face Hub?
46. Какие основные операции с изображениями поддерживает OpenCV?
47. Как выполнить детекцию лиц с помощью каскадов Хаара в OpenCV?
48. Что такое аугментация данных и какие трансформации предоставляет Torchvision?
49. Как загрузить предобученную модель ResNet из Torchvision и адаптировать её под новый датасет?

50. В чём разница между Pillow и OpenCV для обработки изображений?
51. Какие задачи решает библиотека NLTK, а какие – spaCy?
52. Что такое стемминг и лемматизация? В чём разница?
53. Как обучить модель Word2Vec с помощью Gensim и для чего она используется?
54. Что такое тематическое моделирование и как LDA помогает в его реализации?
55. Как выполнить выделение именованных сущностей (NER) с помощью spaCy?
56. Что такое графовые нейронные сети (GNN) и для каких задач они применяются?
57. Как создать граф в NetworkX и визуализировать его?
58. Что такое GCN (Graph Convolutional Network) и как она работает?
59. Какие библиотеки для глубокого обучения на графах вы знаете? В чём их отличия?
60. Для чего используются алгоритмы центральности в анализе графов?
61. Что такое MLflow и какие компоненты входят в его состав?
62. Как DVC помогает управлять версиями данных в ML-проектах?
63. Какие задачи решает Optuna и как он выполняет оптимизацию гиперпараметров?
64. Для чего используется Weights & Biases (wandb) в ML-экспериментах?
65. Как организовать воспроизводимость эксперимента с помощью комбинации Git, DVC и MLflow?
66. Как создать REST API для модели с использованием FastAPI?
67. Что такое ONNX и для чего нужна конвертация моделей в этот формат?
68. Как упаковать ML-приложение в Docker-контейнер?
69. В чём разница между TensorFlow Serving и TorchServe?
70. Как выполнить инференс модели ONNX Runtime и сравнить его скорость с исходным фреймворком?
71. Какие возможности предоставляет Kaggle для обучения и соревнований?
72. Как развернуть демонстрационное приложение на Hugging Face Spaces?
73. Какие преимущества у Yandex DataSphere для российских пользователей?
74. Что такое AutoML в Google Vertex AI и для каких задач он подходит?
75. Сравните облачные платформы для ML по критериям стоимости и функциональности.
76. Что такое RAG (Retrieval-Augmented Generation) и для решения каких проблем LLM он предназначен?
77. Как работает LangChain и какие компоненты входят в его экосистему?
78. Что такое векторная база данных и какие задачи она решает?
79. Какие векторные базы данных вы знаете? В чём отличия Chroma, FAISS и Qdrant?
80. Как оценить качество RAG-пайплайна? Какие метрики используются?

4.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся

Самостоятельная работа обучающихся обеспечивается следующими учебно-методическими материалами:

1. Указаниями в рабочей программе по дисциплине (п.4.1.)
2. Лекционные материалы в составе учебно-методического комплекса по дисциплине
3. Заданиями и методическими рекомендациями по организации самостоятельной работы обучающихся в составе учебно-методического комплекса по дисциплине.
4. Глоссарием по дисциплине в составе учебно-методического комплекса по дисциплине.

Раздел 5. Фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся

Фонд оценочных средств по дисциплине представляет собой совокупность контролирующих материалов, предназначенных для измерения уровня достижения обучающимися установленных результатов образовательной программы. ФОС по дисциплине используется при проведении оперативного контроля и промежуточной аттестации обучающихся. Требования к структуре и содержанию ФОС дисциплины регламентируются Положением о фонде оценочных материалов по программам высшего образования – программам бакалавриата, магистратуры.

5.1. Паспорт фонда оценочных средств

Очная форма обучения (полный срок)

№	Контролируемые разделы (темы) дисциплины	Оценочные средства			
		Л	ПЗ (ЛЗ, СЗ)	СРО	Код индикатора и дескриптора достижения компетенций
1	*2*	*3*	*4*	*5*	*6*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	УО	ЗЗ, МШ	ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	УО	ЗЗ, Д	ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	УО	ЗЗ, Д, МШ	ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	УО	ЗЗ, Д, МП	ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
5	Scikit-learn: унифицированный API для классического машинного обучения	УО	ЗЗ, МШ	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	УО	ЗЗ, Д	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	УО	ЗЗ, МШ	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	УО	ЗЗ, Д	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	УО	ЗЗ, МШ	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision	УО	ЗЗ, Д	ПРВ	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim	УО	ЗЗ, Д, МШ	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL	УО	ЗЗ, Д, МП	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna	УО	ЗЗ, МШ	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe	УО	ЗЗ, Д	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI	УО	ЗЗ, МШ	ПРВ	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
16	Большие языковые модели и RAG-	УО	ЗЗ, Д	ПРВ	ИД-5 ОПК-2.1

	пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)				ИД-6 ОПК-2.2

Очно-заочная форма обучения (полный срок)

№	Контролируемые разделы (темы) дисциплины	Оценочные средства			
		Л	ПЗ (ЛЗ, СЗ)	СРО	Код индикатора и дескриптора достижения компетенций
1	*2*	*3*	*4*	*5*	*6*
1	Обзор экосистемы ИИ: языки программирования (Python, R, Julia), среда разработки (Jupyter, VS Code, Colab)	УО		ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
2	NumPy и SciPy: фундамент численных вычислений и научных расчётов	УО		ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
3	Pandas и Polars: высокопроизводительная обработка и анализ табличных данных	УО		ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
4	Визуализация данных и результатов моделей: Matplotlib, Seaborn, Plotly	УО		ПРВ	ИД-1 ОПК-2.1 ИД-2 ОПК-2.2
5	Scikit-learn: унифицированный API для классического машинного обучения	УО		ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
6	XGBoost, LightGBM, CatBoost: градиентный бустинг на структурированных данных	УО		ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
7	PyTorch: динамические вычислительные графы и исследовательские прототипы	УО		ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
8	TensorFlow и Keras: промышленное развёртывание и высокоуровневый API	УО	ЗЗ, МШ	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
9	Hugging Face Transformers: предобученные модели (BERT, GPT) и их тонкая настройка	УО	ЗЗ, МШ	ПРВ	ИД-2 ОПК-2.2 ИД-3 ОПК-2.1
10	Библиотеки для компьютерного зрения: OpenCV, Pillow, Torchvision		ЗЗ, Д	ПРВ	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
11	Библиотеки для обработки естественного языка: NLTK, spaCy, Gensim		ЗЗ, Д, МШ	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
12	Инструменты для работы с графовыми данными: NetworkX, PyTorch Geometric, DGL		ЗЗ, Д, МП	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
13	MLOps и управление экспериментами: MLflow, DVC, Weights & Biases, Optuna		ЗЗ, МШ	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
14	Развёртывание моделей: FastAPI, ONNX, Docker, TensorFlow Serving, TorchServe		ЗЗ, Д	ПРВ	ИД-3 ОПК-2.1 ИД-4 ОПК-2.2
15	Платформы и сервисы для ИИ-решений: Kaggle, Hugging Face Spaces, Yandex DataSphere, Google Vertex AI		ЗЗ, МШ	ПРВ	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2
16	Большие языковые модели и RAG-пайплайны: LangChain, LlamaIndex, векторные базы данных (Chroma, FAISS, Qdrant)		ЗЗ, МШ	ПРВ	ИД-5 ОПК-2.1 ИД-6 ОПК-2.2

Условные обозначения оценочных средств (Столбцы 3, 4, 5):

ЗЗ – защита выполненных заданий (творческих, расчетных и т.д.), представление презентаций;

ПРВ – проверка рефератов, отчетов, рецензий, аннотаций, конспектов, графического матери-

ала, эссе, переводов, решений заданий, выполненных заданий в электронном виде и т.д.;

МШ – Метод мозгового штурма;

Д – Дискуссия, полемика, диспут, дебаты;

МП – Метод проектов.

5.2. Тематика письменных работ обучающихся

1. Сравнительный анализ производительности NumPy и чистого Python при выполнении матричных операций.
2. Обзор и сравнение методов обработки пропущенных значений в Pandas.
3. Применение группировки и сводных таблиц для анализа продаж интернет-магазина.
4. Визуализация многомерных данных с помощью Seaborn: возможности и ограничения.
5. Создание интерактивных дашбордов с использованием Plotly.
6. Сравнение методов масштабирования данных (StandardScaler, MinMaxScaler, RobustScaler).
7. Построение пайплайнов предобработки и моделирования с Scikit-learn.
8. Сравнение алгоритмов классификации на несбалансированных датасетах.
9. Оптимизация гиперпараметров градиентного бустинга с использованием Optuna.
10. Сравнение XGBoost, LightGBM и CatBoost на структурированных данных.
11. Transfer learning с использованием предобученных моделей Torchvision.
12. Fine-tuning BERT для задачи классификации текстов.
13. Генерация текста с использованием GPT-2: параметры и качество.
14. Обнаружение объектов на изображениях с использованием OpenCV и каскадов Хаара.
15. Применение сверточных нейронных сетей для классификации изображений с использованием PyTorch.
16. Сравнение динамических и статических вычислительных графов на примере PyTorch и TensorFlow 1.x.
17. Построение и обучение рекуррентных нейронных сетей (LSTM) для прогнозирования временных рядов.
18. Использование библиотеки spaCy для извлечения именованных сущностей из текстов.
19. Тематическое моделирование корпуса текстов с помощью LDA в Gensim.
20. Сравнение стемминга и лемматизации при предобработке текстов.
21. Классификация узлов графа с использованием GCN в PyTorch Geometric.
22. Анализ социальных сетей с помощью NetworkX: центральности и кластеризация.
23. Применение графовых нейронных сетей для предсказания свойств молекул.
24. Организация воспроизводимых экспериментов с использованием DVC и MLflow.
25. Логирование и визуализация экспериментов с Weights & Biases.
26. Автоматический подбор гиперпараметров нейронных сетей с Optuna.
27. Создание REST API для модели машинного обучения с использованием FastAPI.
28. Контейнеризация ML-приложения с помощью Docker.
29. Конвертация модели PyTorch в ONNX и ускорение инференса с ONNX Runtime.
30. Сравнение TensorFlow Serving и TorchServe для развертывания моделей в продакшене.
31. Разработка и развертывание приложения на Hugging Face Spaces с использованием Gradio.
32. Использование Kaggle для участия в соревнованиях по машинному обучению.
33. Построение RAG-пайплайна с использованием LangChain и Chroma.
34. Сравнение векторных баз данных Chroma, FAISS и Qdrant для задач поиска.
35. Интеграция LLM с внешними инструментами с использованием LangChain Agents.
36. Создание вопросно-ответной системы по документам с LlamaIndex.
37. Обработка и анализ временных рядов с использованием Pandas и Statsmodels.
38. Прогнозирование временных рядов с помощью LSTM и Prophet: сравнительный анализ.
39. Обнаружение аномалий в данных с использованием изолирующего леса и автоэнкодеров.
40. Применение методов понижения размерности (PCA, t-SNE, UMAP) для визуализации

данных.
41. Создание рекомендательной системы на основе коллаборативной фильтрации с использованием Surprise.
42. Реализация алгоритма K-средних с нуля на NumPy и сравнение с реализацией Scikit-learn.
43. Балансировка классов с использованием SMOTE и других методов ресемплинга.
44. Построение ансамблевых моделей (стекинг, блендинг) для повышения точности прогнозов.
45. Интерпретация моделей машинного обучения с использованием SHAP и LIME.
46. Автоматическое создание признаков (feature engineering) с использованием Featuretools.
47. Работа с большими данными: использование Dask или Polars для обработки датасетов, не помещающихся в память.
48. Реализация сверточной нейронной сети для распознавания рукописных цифр MNIST с нуля на NumPy.
49. Применение библиотеки PyTorch Lightning для структурирования обучающего кода.
50. Анализ тональности текстов с использованием моделей Transformer (ruBERT).

5.3. Перечень вопросов промежуточной аттестации по дисциплине

Вопросы к экзамену:

1. Какие языки программирования наиболее популярны в области ИИ и почему Python является основным?
2. В чём преимущества использования Google Colab перед локальной средой разработки?
3. Что такое виртуальное окружение в Python и зачем оно нужно?
4. Какие основные отличия между Jupyter Notebook и VS Code как средами разработки для ИИ?
5. Что такое массив ndarray в NumPy и чем он отличается от списка Python?
6. Что такое векторизация вычислений в NumPy и каковы её преимущества?
7. Сформулируйте правила трансляции (broadcasting) в NumPy.
8. Какие задачи можно решать с помощью модулей SciPy (optimize, integrate, stats, sparse)?
9. Что такое Series и DataFrame в Pandas? В чём их отличие?
10. Как выполнить фильтрацию строк DataFrame по условию с использованием булевой маски?
11. Какие методы Pandas используются для обнаружения и заполнения пропусков?
12. В чём разница между методами merge и concat в Pandas?
13. Каковы преимущества Polars перед Pandas для работы с большими объёмами данных?
14. Какие типы графиков можно построить с помощью Matplotlib и для каких задач?
15. В чём преимущества Seaborn перед Matplotlib для статистической визуализации?
16. Как создать интерактивный график с Plotly и в чём его преимущества?
17. Как устроен единый API Scikit-learn (fit, predict, transform)?
18. Какие метрики используются для оценки качества классификации и регрессии?
19. Что такое кросс-валидация и зачем она нужна?
20. Как выполнить подбор гиперпараметров с помощью GridSearchCV?
21. Что такое Pipeline в Scikit-learn и какие преимущества он даёт?
22. В чём основное отличие градиентного бустинга от случайного леса?
23. Какие ключевые параметры влияют на качество модели XGBoost?
24. В чём преимущества LightGBM и CatBoost перед XGBoost?
25. Как CatBoost автоматически обрабатывает категориальные признаки?
26. В чём основное отличие PyTorch от TensorFlow?
27. Что такое динамический вычислительный граф и какие преимущества он даёт?
28. Как работает автоматическое дифференцирование (autograd) в PyTorch?
29. Как сохранить и загрузить обученную модель PyTorch?

30. Какие типы API предоставляет Keras (Sequential, Functional, Subclassing)?
31. Что такое transfer learning и как его реализовать в Keras?
32. Для чего используются callbacks EarlyStopping и ModelCheckpoint?
33. Каковы преимущества TensorFlow Serving для промышленного развёртывания?
34. Что такое pipeline в библиотеке Transformers и для каких задач он используется?
35. Как выполнить тонкую настройку (fine-tuning) BERT для задачи классификации текстов?
36. Какие параметры влияют на качество генерации текста в GPT (temperature, top_k, top_p)?
37. Как загрузить предобученную русскоязычную модель из Hugging Face Hub?
38. Какие основные операции с изображениями поддерживает OpenCV?
39. Как выполнить детекцию лиц с помощью каскадов Хаара в OpenCV?
40. Что такое аугментация данных и какие трансформации предоставляет Torchvision?
41. Как загрузить предобученную модель ResNet из Torchvision и адаптировать её под новый датасет?
42. Какие задачи решает библиотека NLTK, а какие – spaCy?
43. Что такое стемминг и лемматизация? В чём разница?
44. Как обучить модель Word2Vec с помощью Gensim и для чего она используется?
45. Что такое графовые нейронные сети (GNN) и для каких задач они применяются?
46. Как создать граф в NetworkX и визуализировать его?
47. Что такое GCN (Graph Convolutional Network) и как она работает?
48. Какие библиотеки для глубокого обучения на графах вы знаете? В чём их отличия?
49. Что такое MLflow и какие компоненты входят в его состав?
50. Как DVC помогает управлять версиями данных в ML-проектах?
51. Какие задачи решает Optuna и как он выполняет оптимизацию гиперпараметров?
52. Для чего используется Weights & Biases (wandb) в ML-экспериментах?
53. Как создать REST API для модели с использованием FastAPI?
54. Что такое ONNX и для чего нужна конвертация моделей в этот формат?
55. Как упаковать ML-приложение в Docker-контейнер?
56. Какие возможности предоставляет Kaggle для обучения и соревнований?
57. Как развернуть демонстрационное приложение на Hugging Face Spaces?
58. Что такое RAG (Retrieval-Augmented Generation) и для решения каких проблем LLM он предназначен?
59. Как работает LangChain и какие компоненты входят в его экосистему?
60. Что такое векторная база данных и какие задачи она решает? Какие векторные базы данных вы знаете?

Раздел 6. Оценочные средства промежуточной аттестации (с ключами)

1. Какая библиотека Python является стандартом для работы с многомерными массивами и векторизованными вычислениями?
А) Pandas
Б) NumPy
В) Matplotlib
Г) Scikit-learn
Правильный ответ: Б

2. Какая функция в Pandas используется для просмотра первых 5 строк DataFrame?
А) tail()
Б) info()
В) head()
Г) describe()
Правильный ответ: В

3. Какой метод в Scikit-learn используется для разделения данных на обучающую и тестовую выборки?
А) cross_val_score
Б) GridSearchCV
В) train_test_split
Г) KFold
Правильный ответ: В

4. Какая метрика качества модели классификации вычисляется как отношение правильно предсказанных положительных объектов ко всем объектам, предсказанным как положительные?
А) Recall
Б) Precision
В) Accuracy
Г) F1-score
Правильный ответ: Б

5. Какой алгоритм градиентного бустинга автоматически обрабатывает категориальные признаки без необходимости в OneHotEncoding?
А) XGBoost
Б) LightGBM
В) CatBoost
Г) AdaBoost
Правильный ответ: В

6. В чём заключается основное отличие PyTorch от TensorFlow 1.x?
А) PyTorch поддерживает только CPU
Б) PyTorch использует статические вычислительные графы
В) PyTorch использует динамические вычислительные графы
Г) PyTorch не поддерживает нейронные сети
Правильный ответ: В

7. Какая библиотека предоставляет высокоуровневый API для работы с предобученными трансформерами (BERT, GPT)?
А) Scikit-learn
Б) Hugging Face Transformers
В) NLTK
Г) Gensim
Правильный ответ: Б

8. Какой метод в OpenCV используется для детекции лиц с использованием предобученных каскадов?
А) detectMultiScale
Б) findContours
В) Canny
Г) HoughCircles
Правильный ответ: А

9. Какая библиотека предназначена для тематического моделирования и обучения векторных представлений слов (Word2Vec)?
А) NLTK
Б) spaCy
В) Gensim

Г) Transformers
Правильный ответ: В

10. Какой класс в PyTorch используется для создания датасетов с возможностью загрузки по батчам?
А) torch.Tensor
Б) torch.nn.Module
В) torch.utils.data.Dataset
Г) torch.autograd.Function
Правильный ответ: В

11. Какая библиотека используется для создания интерактивных веб-дашбордов с визуализацией данных?
А) Matplotlib
Б) Seaborn
В) Plotly
Г) Pillow
Правильный ответ: В

12. Какой формат предназначен для обмена моделями между различными фреймворками глубокого обучения?
А) JSON
Б) ONNX
В) HDF5
Г) PKL
Правильный ответ: Б

13. Какой инструмент используется для контроля версий данных и пайплайнов в ML-проектах?
А) Git
Б) DVC
В) MLflow
Г) Docker
Правильный ответ: Б

14. Какая библиотека предназначена для автоматической оптимизации гиперпараметров?
А) GridSearchCV
Б) Optuna
В) RandomizedSearchCV
Г) MLflow
Правильный ответ: Б

15. Что такое RAG (Retrieval-Augmented Generation)?
А) Архитектура для генерации изображений
Б) Архитектура для улучшения ответов LLM через поиск релевантного контекста
В) Метод оптимизации градиентного спуска
Г) Библиотека для работы с графами
Правильный ответ: Б

16. Какая библиотека позволяет создавать цепочки (chains) и агентов для работы с большими языковыми моделями?
А) LlamaIndex
Б) LangChain
В) Transformers

Г) Gensim
Правильный ответ: Б

17. Какая функция в NumPy создаёт массив из равномерно распределённых чисел в заданном интервале с указанием количества элементов?
А) arange
Б) linspace
В) range
Г) random.rand
Правильный ответ: Б

18. Какой метод в Pandas используется для группировки данных с последующей агрегацией?
А) pivot_table
Б) merge
В) groupby
Г) concat
Правильный ответ: В

19. Какая метрика используется для оценки качества регрессионной модели и измеряет долю дисперсии, объясняемую моделью?
А) MSE
Б) MAE
В) RMSE
Г) R^2
Правильный ответ: Г

20. Какой параметр в XGBoost отвечает за скорость обучения (шаг градиентного спуска)?
А) n_estimators
Б) max_depth
В) learning_rate
Г) subsample
Правильный ответ: В

21. Какой слой в нейронных сетях помогает бороться с переобучением путём случайного отключения нейронов?
А) BatchNormalization
Б) Dropout
В) Dense
Г) Conv2D
Правильный ответ: Б

22. Какая библиотека используется для построения синтаксических деревьев зависимостей и выделения именованных сущностей с высокой производительностью?
А) NLTK
Б) spaCy
В) Gensim
Г) TextBlob
Правильный ответ: Б

23. Какой алгоритм в NetworkX используется для поиска кратчайшего пути между двумя вершинами?
А) bfs_tree
Б) shortest_path

В) connected_components
Г) pagerank
Правильный ответ: Б

24. Какая платформа предоставляет бесплатные GPU/TPU для выполнения Jupyter ноутбуков и участвует в соревнованиях по Data Science?
А) Hugging Face Spaces
Б) Google Colab
В) Yandex DataSphere
Г) Kaggle
Правильный ответ: Г

25. Какой компонент MLflow отвечает за логирование параметров, метрик и артефактов экспериментов?
А) MLflow Models
Б) MLflow Projects
В) MLflow Tracking
Г) MLflow Registry
Правильный ответ: В

26. Какая библиотека является высокоуровневым API для TensorFlow и упрощает создание нейронных сетей?
А) PyTorch
Б) Keras
В) JAX
Г) Caffe
Правильный ответ: Б

27. Что такое FAISS?
А) Библиотека для обработки естественного языка
Б) Библиотека для эффективного поиска ближайших соседей в векторных пространствах
В) Фреймворк для глубокого обучения
Г) Платформа для развёртывания моделей
Правильный ответ: Б

28. Какая библиотека используется для создания REST API для моделей машинного обучения с автоматической генерацией документации Swagger?
А) Flask
Б) Django
В) FastAPI
Г) Tornado
Правильный ответ: В

29. Какой метод в OpenCV используется для выделения границ на изображении?
А) GaussianBlur
Б) Canny
В) threshold
Г) findContours
Правильный ответ: Б

30. Какая библиотека является альтернативой Pandas с поддержкой ленивых вычислений и более высокой производительностью на больших данных?
А) Dask
Б) Polars

В) Modin
Г) Vaex
Правильный ответ: Б

Раздел 7. Перечень учебной литературы, необходимой для освоения дисциплины

7.1. Основная литература

1. Посполит А.В. Основы искусственного интеллекта в примерах на Python : самоучитель. 2-е изд., перераб. и доп. Санкт-Петербург : БХВ-Петербург, 2025. 448 с. (Серия «Самоучитель»). ISBN 978-5-9775-1818-5.
2. Ховард Дж., Гуггер С. Глубокое обучение с fastai и PyTorch: минимум формул, минимум кода, максимум эффективности. Пер. с англ. Д. Брайт. Санкт-Петербург : Питер, 2025. 621 с. (Серия «Бестселлеры O'Reilly»). ISBN 978-5-4461-1475-7.
3. Воронов М.В., Пименов В.И., Небаев И.А. Системы искусственного интеллекта : учебник и практикум для вузов. 2-е изд., перераб. и доп. Москва : Юрайт, 2025. 268 с. (Высшее образование). ISBN 978-5-534-17032-0.
4. Рассел С., Норвиг П. Искусственный интеллект: современный подход. 4-е изд. Москва : Вильямс, 2021. 1408 с. ISBN 978-5-907144-96-7.
5. Колмогорова С.С. Обработка данных алгоритмами искусственного интеллекта в системе интернета вещей : учебное пособие для вузов. 3-е изд., стер. Санкт-Петербург : Лань, 2025. 104 с. ISBN 978-5-507-53069-4.

7.2. Дополнительная литература

1. Китов В.В. Глубокое машинное обучение : онлайн-учебник. 2025. Режим доступа: https://deepmachinelearning.ru.
2. Мишра П. Объяснимые модели искусственного интеллекта на Python. Модель искусственного интеллекта. Объяснения с использованием библиотек, расширений и фреймворков на основе языка Python. Пер. с англ. С.В. Минца. Москва : ДМК Пресс, 2022. 298 с. ISBN 978-5-93700-124-5.
3. Воронов М.В., Пименов В.И., Небаев И.А. Системы искусственного интеллекта : учебник и практикум для вузов. 2-е изд., перераб. и доп. Москва : Юрайт, 2024. 267 с.
4. Вандер Плас Дж. Python для сложных задач: наука о данных и машинное обучение. Санкт-Петербург : Питер, 2018. 576 с. ISBN 978-5-496-03068-9.
5. Жерон О. Прикладное машинное обучение с помощью Scikit-Learn, Keras и TensorFlow. 3-е изд. Москва : Диалектика, 2021. 848 с. ISBN 978-5-907203-94-7.

7.3. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Китов В.В. Онлайн-учебник по машинному и глубокому обучению DeepMachineLearning.ru. Режим доступа: https://deepmachinelearning.ru (дата обращения: 09.04.2026).
2. Электронно-библиотечная система «Лань». Раздел «Искусственный интеллект». Режим доступа: https://e.lanbook.com (дата обращения: 09.04.2026).
3. Образовательная платформа «Юрайт». Раздел «Системы искусственного интеллекта». Режим доступа: https://urait.ru (дата обращения: 09.04.2026).
4. Microsoft Learn. Учебники по искусственному интеллекту и машинному обучению. Режим доступа: https://learn.microsoft.com/ru-ru/azure/databricks/machine-learning/ai-ml-tutorials (дата обращения: 09.04.2026).

5. GitHub. Подборка бесплатных ресурсов по Data Science и Machine Learning (ds_v1). Режим доступа: https://github.com/ELMedoedo/ds_v1 (дата обращения: 09.04.2026).

Раздел 8. Материально-техническая база и информационные технологии

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине:

Материально-техническое обеспечение дисциплины **«Инструменты решения задач искусственного интеллекта»** включает в себя учебные аудитории для проведения занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, самостоятельной работы обучающихся.

Учебные аудитории укомплектованы специализированной мебелью и техническими средствами обучения. Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети Интернет.

Дисциплина может реализовываться с применением дистанционных технологий обучения. Специфика реализации дисциплины с применением дистанционных технологий обучения устанавливается дополнением к рабочей программе. В части не противоречащей специфике, изложенной в дополнении к программе, применяется настоящая рабочая программа.

Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине с применением дистанционных образовательных технологий включает в себя:

Компьютерная техника, расположенная в учебном корпусе Института (ул. Качинцев, 63, кабинет Центра дистанционного обучения):

1. Intel i 3 3.4Ghz\ОЗУ 4Gb\500GB\RadeonHD5450
2. Intel PENTIUM 2.9GHz\ОЗУ 4GB\500GB

3 личные электронные устройства (компьютеры, ноутбуки, планшеты и иное), а также средства связи преподавателей и студентов.

Информационные технологии, необходимые для осуществления образовательного процесса по дисциплине с применением дистанционных образовательных технологий включают в себя:

- система дистанционного обучения (СДО) (Learning Management System) (LMS) Moodle (Modular Object-Oriented Dynamic Learning Environment);
- электронная почта;
- система компьютерного тестирования;
- Цифровой образовательный ресурс IPR SMART;
- система интернет-связи skype;
- телефонная связь;
- ПО для организации конференций.

Обучение обучающихся инвалидов и обучающихся с ограниченными возможностями здоровья осуществляется посредством применения специальных технических средств в зависимости от вида нозологии.

При проведении учебных занятий по дисциплине используются мультимедийные комплексы, электронные учебники и учебные пособия, адаптированные к ограничениям здоровья обучающихся.

Лекционные аудитории оборудованы мультимедийными кафедрами, подключенными к звуковым колонкам, позволяющими усилить звук для категории слабослышащих обучающихся, а также проекционными экранами, которые увеличивают изображение в несколько раз и позволяют воспринимать учебную информацию обучающимся с нарушениями зрения.

При обучении лиц с нарушениями слуха используется усилитель слуха для слабослышащих людей Cyber Ear модель HAP-40, помогающий обучаемым лучше воспринимать учебную информацию.

Обучающиеся с ограниченными возможностями здоровья, обеспечены печатными и электронными образовательными ресурсами (программы, учебники, учебные пособия, материалы для самостоятельной работы и т.д.) в формах, адаптированных к ограничениям их здоровья и восприятия информации:

для лиц с нарушениями зрения:

- в форме электронного документа;
- в форме аудиофайла;

для лиц с нарушениями слуха:
- в печатной форме;
- в форме электронного документа;

для лиц с нарушениями опорно-двигательного аппарата:
- в печатной форме;
- в форме электронного документа;
- в форме аудиофайла.

Раздел 9. Методические указания для обучающихся по освоению дисциплины

Дисциплина включает практические занятия, самостоятельную работу обучающегося.

В ходе изучения дисциплины «Инструменты решения задач искусственного интеллекта» перед обучающимися стоит задача не только закрепить знания о сложных информационных явлениях, о чем свидетельствует содержание тематического плана, глубоко разобраться в объемном учебном материале, но и сформировать у себя на основе полученных компьютерных знаний соответствующие профессионально важные качества.

Практические занятия – один из самых эффективных видов учебных занятий, на которых обучающиеся учатся творчески работать с различной информацией, являются также действенной формой активизации самостоятельной работы обучающихся.

Целью практических занятий является закрепление полученных в ходе лекций, а также в ходе самостоятельной работы над учебной и специальной литературой, знаний, умений и навыков. На практических занятиях особо обращается внимание на умение обучающихся проявлять элементы творчества в процессе самостоятельной работы, применять полученные знания на практике.

Практические занятия занимают центральное место в учебном процессе, так как позволяют на завершающем этапе усвоения материала, после прослушанной лекции и самостоятельного поиска дополнительных сведений по рассматриваемой проблематике, окончательно уточнить, сформировать свои позиции в ходе работы в составе учебной группы.

Основное в подготовке и проведении практикума – это самостоятельная работа обучающегося над изучением темы лекционного материала. Практические занятия проводятся по специальным планам – заданиям, которые содержатся в материалах, подготовленных на кафедре. Обучающийся обязан точно знать план занятия либо конкретное задание к нему.

При подготовке к практическим занятиям следует чаще обращаться к справочной литературе, полнее использовать консультации (групповые и индивидуальные, устные и письменные) с преподавателями, которые читают лекции и проводят практикумы.

Таким образом, в процессе подготовке к практическому занятию рекомендуется:

- ознакомиться с вопросами плана;
- прочитать конспект лекции по изучаемой теме;
- прочитать соответствующие главы учебников, статьи;
- просмотреть перечень научных источников, предлагаемых в рабочей программе, выбрав несколько из них для углубленного изучения данной темы.

По каждому практическому заданию обучающиеся отчитываются преподавателю, оформляя письменный отчёт, в котором сохраняют результаты своей работы в виде файлов. Результаты выполнения практических заданий оцениваются с учетом теоретических знаний по соответствующим вопросам дисциплины и уровнем владения практическими навыками при работе на компьютере.

Для углубленного изучения и освоения материала целесообразно выполнение практических работ, наряду с другими различными формами обучения обучающихся: тесты, задачи, упражнения, которые используются при проведении практических занятий, выполнении контрольных и аудиторных работ, а также при самостоятельном изучении данной дисциплины.

Одним из наиболее интенсивных способов изучения дисциплины является самостоятельное выполнение практических работ, на которых вырабатываются навыки по дисциплине «Инструменты решения задач искусственного интеллекта».

СРО позволяет глубже освоить теоретические и практические вопросы, понять принципы дисциплины «Инструменты решения задач искусственного интеллекта».

Основными задачами организации процесса самостоятельной работы по дисциплине являются:

- приобретение знаний по теоретическим основам дисциплины «Инструменты решения задач искусственного интеллекта», являющихся дополнением к материалу лекционных аудиторных занятий;
- приобретение практических навыков по дисциплине «Инструменты решения задач искусственного интеллекта».

Основные формы реализации СРО – изучение учебно-методической литературы по дисци-

плине «Инструменты решения задач искусственного интеллекта». В качестве базовой литературы можно использовать учебники и учебные пособия, согласно приведенному списку в разделе 6 рабочей программы, а также любые другие источники информации, такие как электронные учебники, обучающие и энциклопедические сайты, публикации журналов и конференций.

Обучающийся допускается к зачетному занятию по результатам успешного выполнения всех практических заданий и самостоятельной работы.

Учебно-методическое издание

Рабочая программа учебной дисциплины

Инструменты решения задач искусственного интеллекта

(Наименование дисциплины в соответствии с учебным планом)

Скоробогатченко Дмитрий Анатольевич

(Фамилия, Имя, Отчество составителя)