Федеральное государственное автономное образовательное учреждение высшего образования
«Российский университет дружбы народов имени Патриса Лумумбы»

Факультет искусственного интеллекта
(наименование основного учебного подразделения (ОУП)-разработчика ОП ВО)

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

МАССОВО-ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ В МАШИННОМ ОБУЧЕНИИ (GPU)
(наименование дисциплины/модуля)

Рекомендована МССН для направлений подготовки:

02.03.02 ФУНДАМЕНТАЛЬНАЯ ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ;
09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА
(код и наименование направления подготовки/специальности)

Освоение дисциплины ведется в рамках реализации основной профессиональной образовательной программы высшего образования (ОП ВО):

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ: РАЗРАБОТКА И ОБУЧЕНИЕ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ
(наименование (профиль/специализация) ОП ВО)

2026 г.

1. ЦЕЛЬ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Дисциплина «Массово-параллельные вычисления в машинном обучении (GPU)» входит в программу бакалавриата «Искусственный интеллект: разработка и обучение интеллектуальных систем» по направлениям подготовки 02.03.02 Фундаментальная информатика и информационные технологии и 09.03.03 Прикладная информатика, и изучается в 6 семестре 3 курса. Дисциплину реализует Кафедра прикладного искусственного интеллекта. Дисциплина состоит из 3 разделов и 34 тем и направлена на изучение архитектуры графических процессоров (GPU) и принципов массово-параллельных вычислений в применении к задачам машинного обучения: модели SIMT и иерархии памяти GPU, программирования ядер CUDA (Numba, CuPy, PyCUDA), оптимизации вычислений на GPU (coalescence, occupancy, shared memory), использования PyTorch CUDA API для обучения нейронных сетей, методов оптимизации GPU-обучения (mixed precision, gradient accumulation, gradient checkpointing), профилирования и устранения узких мест, стратегий распределённого обучения на нескольких GPU (DataParallel, DistributedDataParallel, FSDP), а также методов оптимизации инференса (TensorRT, ONNX Runtime, квантизация).

Целью освоения дисциплины является формирование у студентов системных знаний об архитектуре GPU и практических навыков использования массово-параллельных вычислений для ускорения обучения и инференса моделей машинного обучения, включая способность программировать вычислительные ядра, переносить вычисления на GPU средствами PyTorch, профилировать и оптимизировать производительность, применять стратегии распределённого обучения на нескольких GPU, а также проектировать инфраструктуру GPU-вычислений для ИИ-систем с учётом требований к производительности и стоимости.

2. ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Освоение дисциплины «Массово-параллельные вычисления в машинном обучении (GPU)» направлено на формирование у обучающихся следующих компетенций (части компетенций):

Таблица 2.1. Перечень компетенций, формируемых у обучающихся при освоении дисциплины (результаты освоения дисциплины)

Шифр	**Компетенция**	**Индикаторы достижения компетенции** (в рамках данной дисциплины)
ОПК-2	Способен понимать принципы работы современных информационных технологий и применять компьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности	ОПК-2.1 Знает принципы работы современных информационных технологий, включая технологии машинного обучения, облачных вычислений, высокопроизводительных вычислений (HPC) и параллельного программирования; ОПК-2.3 Владеет навыками использования вычислительных методов, включая массово-параллельные вычисления на GPU, для обучения и развёртывания моделей ИИ;
ОПК-5	Способен инсталлировать и сопровождать программное и аппаратное обеспечение информационных систем и систем ИИ, в том числе отечественного происхождения, с учётом требований информационной безопасности	ОПК-5.3 Владеет навыками практической работы с инфраструктурой ИИ-систем (облачные платформы, серверы GPU, системы хранения данных), включая мониторинг, обновление и обеспечение отказоустойчивости;
ПК-1	Способен анализировать	ПК-1.1 Анализирует возможности реализации

Шифр	Компетенция	Индикаторы достижения компетенции (в рамках данной дисциплины)
	требования к программному обеспечению систем ИИ, разрабатывать технические спецификации и техническое задание на систему	функциональных и нефункциональных требований к ПО систем ИИ, выявляет противоречия и ограничения;
ПК-2	Способен проектировать архитектуру информационных систем с компонентами ИИ, разрабатывать прототипы и базы данных таких систем	ПК-2.1 Проектирует архитектуру ИС с компонентами ИИ, выбирает архитектурные паттерны и технологический стек;
BD-4	Способен применять различные модели и (или) технологии обработки больших данных	BD-4.1 Осуществляет выбор технологий обработки больших данных, приемлемых для создания прикладной системы ИИ с заданными требованиями; BD-4.2 Разрабатывает и отлаживает прикладные решения с элементами ИИ с применением различных технологий обработки данных;
FC-1	Способен проводить передовые исследования в области архитектур, алгоритмов МО, оптимизации и математики	FC-1.3 Развивает методы ускорения обучения;

3. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОП ВО

Дисциплина «Массово-параллельные вычисления в машинном обучении (GPU)» относится к обязательной части блока 1 «Дисциплины (модули)» образовательной программы высшего образования.

В рамках образовательной программы высшего образования обучающиеся также осваивают другие дисциплины и/или практики, способствующие достижению запланированных результатов освоения дисциплины «Массово-параллельные вычисления в машинном обучении (GPU)».

Таблица 3.1. Перечень компонентов ОП ВО, способствующих достижению запланированных результатов освоения дисциплины

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
ОПК-2	Способен понимать принципы работы современных информационных технологий и применять компьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности	История и теория программирования; Параллельное и распределенное программирование; Введение в искусственный интеллект; Программирование на языке Python; Hadoop, SPARK; Методы машинного обучения;	Методы машинного обучения; Нейронные сети;
ОПК-5	Способен инсталлировать и сопровождать программное и аппаратное обеспечение информационных систем и систем ИИ, в том числе отечественного	Эксплуатационная практика (учебная); Технологическая (проектно-технологическая) практика (учебная); Введение в базы данных; Методы разработки решений на	Безопасность систем искусственного интеллекта; MLOps и промышленная разработка систем искусственного интеллекта;

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
	происхождения, с учётом требований информационной безопасности	основе искусственного интеллекта (Git, Docker);	
ПК-1	Способен анализировать требования к программному обеспечению систем ИИ, разрабатывать технические спецификации и техническое задание на систему	Эксплуатационная практика (учебная); Технологическая (проектно-технологическая) практика (учебная); Правоведение; Параллельное и распределенное программирование; Введение в искусственный интеллект; Искусственный интеллект и когнитивная психология; Этика и безопасность использования искусственного интеллекта; Методы машинного обучения; История и теория программирования; Программирование на языке C++; Методы разработки решений на основе искусственного интеллекта (Git, Docker); Введение в базы данных; Онтология и графы знаний;	Преддипломная практика; Технологическая (проектно-технологическая) практика (производственная); Методы машинного обучения; Оптимизация моделей машинного обучения; Безопасность систем искусственного интеллекта; Практическая подготовка на проектах отраслевых индустриальных партнеров; MLOps и промышленная разработка систем искусственного интеллекта; Нейронные сети; Проектирование и разработка систем компьютерного зрения; Практикум по обработке естественного языка (NLP);
ПК-2	Способен проектировать архитектуру информационных систем с компонентами ИИ, разрабатывать прототипы и базы данных таких систем	Программирование на языке C++; Параллельное и распределенное программирование; Методы разработки решений на основе искусственного интеллекта (Git, Docker); Алгоритмы и структуры данных; Hadoop, SPARK; Программирование на языке Python; Введение в базы данных; Онтология и графы знаний; Технологическая (проектно-технологическая) практика (учебная); Эксплуатационная практика (учебная);	MLOps и промышленная разработка систем искусственного интеллекта; Практическая подготовка на проектах отраслевых индустриальных партнеров; Проектирование и разработка систем компьютерного зрения; Практикум по обработке естественного языка (NLP); *Вайб-кодинг***; Преддипломная практика; Технологическая (проектно-технологическая) практика (производственная);
BD-4	Способен применять различные модели и (или) технологии обработки больших данных	Hadoop, SPARK;	MLOps и промышленная разработка систем искусственного интеллекта;
FC-1	Способен проводить передовые исследования в области архитектур, алгоритмов МО, оптимизации и математики	Эксплуатационная практика (учебная); Линейная алгебра; Математический анализ; Теория вероятностей и математическая статистика; Методы машинного обучения; Введение в искусственный интеллект; Численная линейная алгебра; Параллельное и распределенное	Преддипломная практика; Технологическая (проектно-технологическая) практика (производственная); Методы машинного обучения; Оптимизация моделей машинного обучения; Нейронные сети; Практическая подготовка на проектах отраслевых

Шифр	Наименование компетенции	Предшествующие дисциплины/модули, практики*	Последующие дисциплины/модули, практики*
		программирование;	индустриальных партнеров;

* - заполняется в соответствии с матрицей компетенций и СУП ОП ВО
** - элективные дисциплины /практики

4. ОБЪЕМ ДИСЦИПЛИНЫ И ВИДЫ УЧЕБНОЙ РАБОТЫ

Общая трудоемкость дисциплины «Массово-параллельные вычисления в машинном обучении (GPU)» составляет «4» зачетные единицы.

Таблица 4.1. Виды учебной работы по периодам освоения образовательной программы высшего образования для очной формы обучения.

Вид учебной работы	**ВСЕГО,** ак.ч.		**Семестр(-ы)**
			6
Контактная работа, ак.ч.	*68*		*68*
Лекции (**ЛК**)	17		17
Лабораторные работы (**ЛР**)	0		0
Практические/семинарские занятия (**СЗ**)	51		51
Самостоятельная работа обучающихся, ак.ч.	*49*		*49*
Контроль (экзамен/зачет с оценкой), ак.ч.	*27*		*27*
Общая трудоемкость дисциплины	**ак.ч.**	***144***	*144*
	зач.ед.	***4***	*4*

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Таблица 5.1. Содержание дисциплины (модуля) по видам учебной работы

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
Раздел 1	Архитектура GPU и основы CUDA-программирования	1.1	Архитектура GPU и модель массово-параллельных вычислений	Эволюция GPU: от графики к вычислениям общего назначения (GPGPU). Архитектура NVIDIA GPU: SM (Streaming Multiprocessor), ядра CUDA, warp, блок потоков, сетка. Модель SIMT (Single Instruction, Multiple Threads). Иерархия памяти: регистры, shared memory, L1/L2 кэш, глобальная память, константная, текстурная. Пропускная способность памяти и вычислений. Поколения GPU: Volta, Ampere, Hopper. Tensor Cores	ЛК	ОПК-2.1, ГС-1.3
		1.2	Модель программирования CUDA	CUDA: хост (CPU) и устройство (GPU). Ядра (kernels): запуск, конфигурация (grid, block). Индексация потоков: threadIdx, blockIdx, blockDim, gridDim. Синхронизация: __syncthreads. Потоки CUDA (streams): асинхронные операции, overlapping. Управление памятью: cudaMalloc, cudaMemcpy, cudaFree. Unified Memory (обзор). Обработка ошибок CUDA	ЛК	ОПК-2.1, ОПК-2.3
		1.3	Оптимизация GPU-вычислений	Coalescent memory access: выровненный доступ к глобальной памяти. Shared memory: банковые конфликты, тайлинг. Occupancy: соотношение активных варпов к максимальным, факторы (регистры, shared memory, блоки). Divergence: расхождение потоков в warp при ветвлении. Arithmetic intensity и roofline model. Принципы оптимизации: минимизация трансферов CPU↔GPU, максимизация occupancy, coalescence	ЛК	ОПК-2.1, ГС-1.3
		1.4	Практикум: среда разработки и первые GPU-вычисления	Настройка среды: CUDA Toolkit, nvidia-smi, nvcc. Google Colab / облачный GPU. Проверка GPU: torch.cuda.is_available(), device properties. CuPy: GPU-ускоренный аналог NumPy. Практика: поэлементные операции на CuPy, замер ускорения по сравнению с NumPy для различных размеров массивов	СЗ	ОПК-2.3, ОПК-5.3
		1.5	Практикум: написание CUDA-ядер с Numba	Numba CUDA: декоратор @cuda.jit. Запуск ядра: конфигурация grid и block. Индексация потоков. Практика: реализация поэлементного сложения векторов, скалярного	СЗ	ОПК-2.3, ГС-1.3

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				умножения. Замер: Numba CUDA vs. NumPy vs. CuPy. Обработка ошибок и отладка		
		1.6	Практикум: матричные операции на GPU	Наивное матричное умножение: одна нить — один элемент результата. Тайлинг с shared memory: загрузка подматриц в shared memory, синхронизация, вычисление. Сравнение наивного и тайлинг подходов по времени. Сравнение с cuBLAS (через CuPy). Обсуждение: почему библиотеки быстрее ручных ядер	СЗ	ОПК-2.3, FC-1.3
		1.7	Практикум: редукция на GPU	Задача редукции: суммирование элементов массива. Наивная редукция: расхождение потоков. Оптимизированная: последовательная адресация, развёртка последнего warp. Атомарные операции (atomicAdd). Практика: реализация параллельной суммы, сравнение с CuPy sum	СЗ	ОПК-2.3, FC-1.3
		1.8	Практикум: CUDA streams и асинхронные операции	CUDA streams: создание, асинхронный запуск ядер, асинхронное копирование памяти. Overlapping: вычисления + передача данных. События (events) для измерения времени. Практика: конвейеризация обработки нескольких батчей данных с помощью streams	СЗ	ОПК-2.3, FC-1.3
		1.9	Практикум: CuPy для научных вычислений на GPU	CuPy: API, совместимость с NumPy. Операции: линейная алгебра (cupy.linalg), FFT (cupy.fft), рандомизация (cupy.random). Пользовательские ядра: RawKernel, ElementwiseKernel, ReductionKernel. Практика: ускорение научных вычислений (SVD, решение СЛАУ) на GPU	СЗ	FC-1.3, ОПК-2.3
		1.10	Практикум: профилирование GPU-кода	Инструменты: nvidia-smi (мониторинг утилизации), nsys (Nsight Systems: timeline, bottlenecks), ncu (Nsight Compute: анализ ядер, occupancy, memory throughput). torch.cuda.Event для замера времени. Практика: профилирование ядер из предыдущих занятий, выявление bottlenecks, оптимизация	СЗ	ОПК-2.3, ОПК-5.3
		1.11	Практикум: GPU-ускорение обработки данных	RAPIDS cuDF: GPU-ускоренный аналог Pandas. Загрузка CSV, фильтрация, groupby, merge на GPU. cuML: GPU-ускоренные ML-алгоритмы (KMeans, PCA, Random Forest). Практика: сравнение cuDF+cuML vs. Pandas+sklearn по времени на большом датасете	СЗ	BD-4.1, BD-4.2, FC-1.3
		1.12	Практикум: мини-проект — GPU-	Сквозная задача: загрузка большого датасета →	СЗ	BD-4.2,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
			ускоренный пайплайн обработки данных	предобработка на GPU (cuDF) → PCA/KMeans на GPU (cuML) → визуализация результатов. Замер ускорения на каждом этапе. Анализ: где GPU даёт наибольший выигрыш, где overhead передачи данных нивелирует ускорение. Документирование		ГС-1.3, ПК-1.1
Раздел 2	PyTorch на GPU: обучение и оптимизация нейронных сетей	2.1	PyTorch CUDA API: тензоры, устройства, автоградиент	PyTorch тензоры на GPU: .to(device), .cuda(), .cpu(). Операции на GPU: автоматическое выполнение на устройстве тензора. Autograd: backward(), grad, вычислительный граф. Управление памятью: torch.cuda.memory_allocated, memory_reserved, empty_cache. Pin memory для ускорения передачи данных. Контекст torch.no_grad() и torch.inference_mode()	ЛК	ОПК-2.3, ГС-1.3
		2.2	Оптимизация GPU-обучения: mixed precision и gradient accumulation	Mixed precision training: FP32 vs. FP16 vs. BF16. Tensor Cores и поддержка FP16. torch.amp: autocast и GradScaler. Loss scaling для предотвращения underflow. Gradient accumulation: эмуляция большого батча при ограниченной памяти. Gradient checkpointing: пересчёт активаций вместо хранения. Обзор: torch.compile (TorchDynamo, Inductor)	ЛК	ГС-1.3, ОПК-2.3
		2.3	Распределённое обучение на нескольких GPU	Стратегии: Data Parallelism (DP, DDP), Model Parallelism, Pipeline Parallelism, Tensor Parallelism. DataParallel: простой API, GIL-ограничение. DistributedDataParallel (DDP): AllReduce градиентов, NCCL backend, DistributedSampler. FSDP (Fully Sharded Data Parallel): шардирование параметров, градиентов, состояния оптимизатора. DeepSpeed ZeRO (обзор). Выбор стратегии в зависимости от размера модели и числа GPU	ЛК	ГС-1.3, ОПК-2.1, ПК-2.1
		2.4	Практикум: обучение нейронной сети на GPU (PyTorch)	Перенос модели и данных на GPU. DataLoader: pin_memory, num_workers. Цикл обучения: forward → loss → backward → step. Замер времени эпохи: CPU vs. GPU. Визуализация: loss и accuracy по эпохам. Практика: обучение CNN на CIFAR-10 на GPU	СЗ	ОПК-2.3, ГС-1.3
		2.5	Практикум: DataLoader и оптимизация загрузки данных	Bottleneck: CPU-предобработка → GPU простаивает. Параметры DataLoader: num_workers (оптимальный подбор), pin_memory, prefetch_factor, persistent_workers. Профилирование: torch.profiler для анализа времени	СЗ	ГС-1.3, ОПК-2.3

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				загрузки vs. вычислений. Практика: оптимизация DataLoader, замер влияния на throughput (images/sec)		
		2.6	Практикум: mixed precision training	torch.amp.autocast: обёртка forward pass. GradScaler: масштабирование loss, unscale, update. Практика: обучение ResNet-18 на CIFAR-10 с FP32 vs. AMP. Замеры: время эпохи, использование памяти, метрики качества. Обсуждение: когда AMP безопасен, когда возможна потеря качества	СЗ	ГС-1.3, ОПК-2.3
		2.7	Практикум: gradient accumulation и gradient checkpointing	Gradient accumulation: накопление градиентов за N мини-батчей, один шаг оптимизатора. Реализация в цикле обучения. Gradient checkpointing: torch.utils.checkpoint. Практика: обучение большой модели на GPU с ограниченной памятью (эффективный батч 256 при физическом 32)	СЗ	ГС-1.3, ОПК-2.3
		2.8	Практикум: профилирование обучения нейронных сетей	torch.profiler: запись trace, анализ операций (CPU vs. CUDA). TensorBoard plugin: визуализация timeline, kernel analysis. Выявление bottlenecks: data loading, forward, backward, optimizer step. Практика: профилирование обучения ResNet, выявление и устранение узких мест	СЗ	ОПК-5.3, ГС-1.3
		2.9	Практикум: torch.compile и оптимизация графа вычислений	torch.compile: режимы (default, reduce-overhead, max-autotune). Принцип: TorchDynamo (capture graph) + Inductor (optimize + codegen). Ограничения: динамические формы, graph breaks. Практика: сравнение скорости обучения и инференса с/без torch.compile. Анализ graph breaks	СЗ	ГС-1.3, ОПК-2.3
		2.10	Практикум: DistributedDataParallel (DDP)	Настройка DDP: init_process_group (NCCL), DistributedSampler, DistributedDataParallel. Запуск: torchrun. Обучение модели на нескольких GPU (или эмуляция на процессах). Сохранение чекпоинтов в DDP. Практика: обучение ResNet с DDP, замер scaling efficiency	СЗ	ГС-1.3, ОПК-2.3, ПК-2.1
		2.11	Практикум: FSDP — обучение больших моделей	FSDP (torch.distributed.fsdp): шардирование параметров и градиентов. Настройка: sharding_strategy, auto_wrap_policy. Сравнение DDP vs. FSDP по использованию памяти. Практика: обучение модели, не помещающейся в память одного GPU, с помощью FSDP	СЗ	ГС-1.3, ОПК-2.1
		2.12	Практикум: мини-проект — оптимизация	Сквозная задача: обучение модели на датасете →	СЗ	ГС-1.3,

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
			обучения нейросети	профилирование → выявление bottlenecks → применение оптимизаций (AMP, оптимизация DataLoader, torch.compile, gradient accumulation) → замер ускорения на каждом шаге → DDP на нескольких GPU. Документирование: таблица «оптимизация → ускорение → trade-off»		ОПК-5.3, ПК-1.1
Раздел 3	Оптимизация инференса и проектирование GPU-инфраструктуры	3.1	Оптимизация инференса: TensorRT, ONNX, квантизация	Инференс vs. обучение: различия требований. ONNX: экспорт PyTorch модели (torch.onnx.export), ONNX Runtime (ORT): CPU и GPU провайдеры. TensorRT: принцип (graph optimization, kernel fusion, precision calibration). Квантизация: post-training quantization (PTQ), quantization-aware training (QAT). INT8, FP16, INT4. Distillation (обзор). Pruning (обзор). Батчирование запросов при инференсе	ЛК	ГС-1.3, ПК-2.1
		3.2	Проектирование GPU-инфраструктуры для ИИ	Выбор GPU для задачи: потребительские (RTX), серверные (A100, H100), облачные (AWS, GCP, Yandex Cloud). Метрики: TFLOPS, память, пропускная способность, NVLink. Стоимость: покупка vs. аренда, spot instances. Инфраструктура: NVIDIA DGX (обзор), Kubernetes + GPU (обзор). Мониторинг GPU: DCGM, Prometheus + Grafana. Планирование ресурсов для ML-проекта	ЛК	ПК-2.1, ОПК-5.3, ПК-1.1
		3.3	Практикум: экспорт модели в ONNX	torch.onnx.export: трассировка, динамические оси. Проверка: onnx.checker, onnxruntime.InferenceSession. Замер инференса: PyTorch vs. ONNX Runtime (CPU, CUDA). Практика: экспорт ResNet, BERT (обзор) в ONNX, сравнение латентности и throughput	СЗ	ГС-1.3, ОПК-2.3
		3.4	Практикум: TensorRT — оптимизация инференса	Конвертация ONNX → TensorRT (trtexec, Python API). Выбор точности: FP32, FP16, INT8 (калибровка). Профилирование: trtexec --profilingVerbosity. Практика: оптимизация CNN с TensorRT, замер ускорения: PyTorch → ONNX → TensorRT	СЗ	ГС-1.3, ПК-2.1
		3.5	Практикум: квантизация моделей (PyTorch)	Post-training quantization: torch.quantization (dynamic, static). Калибровка: сбор статистик активаций. QAT: Quantization-Aware Training с fake quantize. Практика: квантизация MobileNet, замер: размер модели, латентность, accuracy drop	СЗ	ГС-1.3, ОПК-2.3
		3.6	Практикум: инференс-сервер Triton (обзор)	NVIDIA Triton Inference Server: архитектура (model repository, backend, scheduler). Dynamic batching. Model	СЗ	ПК-2.1, BD-4.1

Номер раздела	Наименование раздела дисциплины	Наименование темы		Содержание темы	Вид учебной работы *	Формируемые индикаторы
				ensemble. Развёртывание в Docker. Практика: развёртывание ONNX-модели в Triton, отправка запросов через HTTP/gRPC, замер throughput при разных batch size		
		3.7	Практикум: профилирование инференса end-to-end	Профилирование полного пайплайна инференса: предобработка (CPU) → передача на GPU → forward pass → постобработка (CPU). Определение bottleneck. Оптимизация: перенос предобработки на GPU (torchvision transforms on CUDA, DALI обзор). Практика: оптимизация пайплайна CV-модели	СЗ	ОПК-5.3, FC-1.3
		3.8	Практикум: облачные GPU и управление ресурсами	Работа с облачными GPU: Google Colab Pro, Yandex Cloud GPU (обзор), AWS EC2 GPU (обзор). Мониторинг: nvidia-smi, gpustat. Экономия: spot/preemptible instances, checkpointing для возобновления обучения. Расчёт стоимости обучения модели. Практика: запуск обучения в облаке с checkpointing	СЗ	ОПК-5.3, ПК-2.1
		3.9	Практикум: расчёт требований к GPU-инфраструктуре	Оценка требований: размер модели (параметры × dtype), размер батча и активаций, overhead оптимизатора. Формулы оценки памяти для обучения и инференса. Оценка времени обучения: FLOPS модели / GPU TFLOPS × overhead. Практика: расчёт GPU-ресурсов для учебного проекта, обоснование выбора GPU	СЗ	ПК-1.1, ПК-2.1, ОПК-5.3
		3.10	Практикум: итоговый проект — оптимизация ML-модели для продакшена	Сквозная задача: обученная модель → профилирование → mixed precision → torch.compile → экспорт в ONNX → TensorRT → квантизация → развёртывание (FastAPI + Docker). Таблица: «этап → латентность → throughput → accuracy → размер модели». Расчёт стоимости инференса. Документирование: техническая спецификация инференс-сервиса. Презентация	СЗ	FC-1.3, ПК-2.1, ПК-1.1, ОПК-5.3

* - заполняется только по **ОЧНОЙ** форме обучения: *ЛК – лекции; ЛР – лабораторные работы; СЗ – практические/семинарские занятия.*

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Таблица 6.1. Материально-техническое обеспечение дисциплины

Тип аудитории	**Оснащение аудитории**	**Специализированное учебное/лабораторное оборудование, ПО и материалы для освоения дисциплины (при необходимости)**
Лекционная	Аудитория для проведения занятий лекционного типа, оснащенная комплектом специализированной мебели; доской (экраном) и техническими средствами мультимедиа презентаций.	
Семинарская	Аудитория для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенная комплектом специализированной мебели и техническими средствами мультимедиа презентаций.	Персональные компьютеры, необходимое ПО
Для самостоятельной работы	Аудитория для самостоятельной работы обучающихся (может использоваться для проведения семинарских занятий и консультаций), оснащенная комплектом специализированной мебели и компьютерами с доступом в ЭИОС.	Персональные компьютеры, необходимое ПО

* - аудитория для самостоятельной работы обучающихся указывается **ОБЯЗАТЕЛЬНО**!

7. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература:

1. Параллельные вычисления на GPU. Архитектура и программная модель CUDA: Учеб. пособие / А. В. Боресков и др. Предисл.: В. А. Садовничий. – 2-е издание. – М.: Издательство Мос ковского университета, 2015. – 336 с., илл. – (Серия «Суперкомпьютерное образование») ISBN 978-5-19-011058-6

Дополнительная литература:

1. Тоуманен, Б. Программирование GPU при помощи Python и CUDA : практическое пособие / Б. Тоуманен ; пер. с англ. А. В. Борескова. - Москва : ДМК Пресс, 2020. - 254 с. - ISBN 978-5-97060-821-0. - Текст : электронный. - URL: https://znanium.ru/catalog/product/1210649

Ресурсы информационно-телекоммуникационной сети «Интернет»:

1. ЭБС РУДН и сторонние ЭБС, к которым студенты университета имеют доступ на основании заключенных договоров

- Электронно-библиотечная система РУДН – ЭБС РУДН https://mega.rudn.ru/MegaPro/Web
- ЭБС «Университетская библиотека онлайн» http://www.biblioclub.ru
- ЭБС «Юрайт» http://www.biblio-online.ru

- ЭБС «Консультант студента» www.studentlibrary.ru
- ЭБС «Знаниум» https://znanium.ru/

2. Базы данных и поисковые системы
   - Sage https://journals.sagepub.com/
   - Springer Nature Link https://link.springer.com/
   - Wiley Journal Database https://onlinelibrary.wiley.com/
   - Наукометрическая база данных Lens.org https://www.lens.org

Учебно-методические материалы для самостоятельной работы обучающихся при освоении дисциплины/модуля:*

1. Курс лекций по дисциплине «Массово-параллельные вычисления в машинном обучении (GPU)».

* - все учебно-методические материалы для самостоятельной работы обучающихся размещаются в соответствии с действующим порядком на странице дисциплины **в ТУИС**!