Федеральное государственное бюджетное образовательное учреждение
высшего образования
Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

УТВЕРЖДАЮ
декан факультета вычислительной
математики и кибернетики

_____/И.А. Соколов /
«__» _октября_____2021г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Робастные модели в машинном обучении»

Уровень высшего образования:

магистратура

Направление подготовки / специальность:

01.04.02 "Прикладная математика и информатика" (3++)

Направленность (профиль) ОПОП:

Искусственный интеллект в кибербезопасности

Форма обучения:

очная

Рабочая программа рассмотрена и утверждена
на заседании Ученого совета факультета ВМК
(протокол №_7_, от 29 сентября 2021 года)

Москва 2021

Рабочая программа дисциплины (модуля) разработана в соответствии с самостоятельно установленным МГУ образовательным стандартом (ОС МГУ) для реализуемых основных профессиональных образовательных программ высшего образования по направлению подготовки 01.04.02 "Прикладная математика и информатика" программы магистратуры в редакции приказа МГУ от 21 декабря 2021 года No 1404.

1. Место дисциплины (модуля) в структуре ОПОП ВО:
Дисциплина (модуль) относится к обязательной части основной профессиональной обра-зовательной программы.

2. Входные требования для освоения дисциплины (модуля), предварительные условия:
учащиеся должны владеть знаниями по математическому анализу, линейной алгебре и теории вероятностей в объеме, соответствующем программе обучения основных образовательных программ бакалавриата по укрупненным группам направлений и специальностей 01.00.00 «Математика и механика», 02.00.00 «Компьютерные и информационные науки» и другим направлениям подготовки бакалавриата.

3. Результаты обучения по дисциплине (модулю), соотнесенные с требуемыми компетенциями выпускников.

Планируемые результаты обучения по дисциплине (модулю)		
Содержание и код компетенции.	**Индикатор (показатель) достижения компетенции**	**Планируемые результаты обучения по дисциплине, сопряженные с индикаторами достижения компетенций**
ОПК-3. Способен разрабатывать математические модели и проводить их анализ при решении задач в области профессиональной деятельности	ОПК-3.1. Знает возможности современных инструментальных средств и систем программирования в области профессиональной деятельности. ОПК-3.2. Умеет проводить сравнительный анализ и осуществлять выбор инструментальных средств для решения задач в области профессиональной деятельности. ОПК-3.3. Имеет практический опыт разработки математических моделей и их анализа при решении задач в области профессиональной деятельности.	ОПК-3.1. З-2 ЗНАТЬ: современные методы построения и анализа математических моделей, возникающих при решении естественнонаучных задач, а также современные методы разработки и реализации алгоритмов их решения. ОПК-3.1. У-1 УМЕТЬ: применять современные методы построения и анализа математических моделей, возникающих при решении естественнонаучных задач, а также современные методы разработки и реализации алгоритмов их решения. ОПК-3.1. В-1 ВЛАДЕТЬ: навыками оптимального выбора современных методов построения и анализа математических моделей, возникающих при решении естественнонаучных задач, а также современных методов разработки и реализации алгоритмов их решения
ОПК-5. Способен разрабатывать алгоритмы и программные средства для решения задач в области создания и применения искусственного интеллекта	ОПК-5.1. Применяет инструментальные среды, программно-технические платформы для решения задач в области создания и применения искусственного интеллекта	ОПК-5.1. З-1. Знает инструментальные среды, программно-технические платформы для решения профессиональных задач ОПК-5.1. У-1. Умеет применять инструменталь-

	ОПК-5.2. Разрабатывает оригинальные программные средства для решения задач в области создания и применения искусственного интеллекта	ные среды, программно-технические платформы для решения профессиональных задач ОПК-5.2. З-1. Знает принципы разработки оригинальных программных средств для решения профессиональных задач ОПК-5.2. У-1. Умеет разрабатывать оригинальные программные средства для решения задач в области создания и применения искусственного интеллекта
ОПК-7. Способен использовать методы научных исследований и математического моделирования в области проектирования и управления системами искусственного интеллекта	ОПК-7.1. Применяет логические методы и приемы научного исследования, методологические принципы современной науки, направления, концепции, источники знания и приемы работы с ними, основные особенности научного метода познания, программно-целевые методы решения научных проблем в профессиональной деятельности	ОПК-7.1. З-1. Знает логические методы и приемы научного исследования; методологические принципы современной науки, направления, концепции, источники знания и приемы работы с ними; основные особенности научного метода познания; программно-целевые методы решения научных проблем; основы моделирования управленческих решений; динамические оптимизационные модели; математические модели оптимального управления для непрерывных и дискретных процессов, их сравнительный анализ; многокритериальные методы принятия решений в профессиональной деятельности ОПК-7.1. У-1. Умеет применять логические методы и приемы научного исследования; методологические принципы современной науки, концепции, источники знания и приемы работы с ними; основные метода научного познания; программно-целевые методы решения научных проблем; основы моделирования управленческих решений; динамические оптимизационные модели; математические модели оптимального управления для непрерывных и дискретных процессов, их сравнительный анализ; многокритериальные методы принятия решений в профессиональной деятельности

4. Объем дисциплины (модуля) составляет 6 з.е., в том числе 108 академических часа, отведенных на контактную работу обучающихся с преподавателем, 108 академических часов на самостоятельную работу обучающихся.

5. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и виды учебных занятий:

Целью курса является обучение слушателей эффективной разработке робастных моделей машинного обучения, для достижения чего необходимо решить следующие задачи:

1. подробно рассмотреть применяемый в данной области математический аппарат;
2. всесторонне ознакомить слушателей с теорией машинного обучения и основными архитектурами ИНС;
3. рассмотреть атаки на модели МО и подходы для защиты от них;
4. представить современные методы оценки робастности моделей МО.

Наименование и краткое содержание разделов и тем дисциплины (модуля), **форма промежуточной аттестации по дисциплине (модулю)**	**Всего (часы)**	В том числе								
		Контактная работа (работа во взаимодействии с преподавателем), часы из них						**Самостоятельная работа обучающегося, часы** из них		
		Занятия лекционного типа	Практические занятия	Групповые консультации	Индивидуальные консультации	Учебные занятия, направленные на проведение текущего контроля успеваемости (коллоквиумы, практические контрольные занятия и др)	**Всего**	Выполнение домашних заданий	Подготовка рефератов и т.п.	**Всего**

1 семестр										
Тема 1. Базовый математический аппарат - линейная алгебра - теория вероятностей и теория информации - численные методы	8	4	-	-	-	-	4	4	-	4
Тема 2. Язык программирования Python	14	-	6	-	-	-	6	8	-	8
Тема 3. Основы машинного обучения - алгоритмы обучения - обучение и недообучение - гипперпараметры и контрольные наборы - оценки, смещения, дисперсия - оценка максимального правдоподобия - байсовская статистика - алгоритмы обучения с учителем - алгоритмы обучения без учителя - стохастический градиентный спуск - постоение алгоритма машинного обучения - глубокое обучение	22	6	8	-	-	-	14	8	-	8
Тема 4. Прикладная библиотека для разработки моделей МО PyTorch	10	2	-	-	-	-	2	8	-	8
Тема 5. Базовые архитектуры ГНС - ИНС прямого распространения - сверточные сети - рекуррентные сети	18	6	4	-	-	-	10	8	-	8

- автоэнкодеры										
6. Промежуточная аттестация – экзамен										
2 семестр										
Тема 7. ГНС для обработки изображений	18	6	6	-	-	-	12	6	-	6
Тема 8. ГНС для обработки звука	18	6	6	-	-	-	12	6	-	6
Тема 9. ГНС для обработки видеоряда	18	6	6	-	-	-	12	6	-	6
Тема 10. ГНС для обработки текстовых данных	18	6	6	-	-	-	12	6	-	6
Тема 11. Базовые принципы создания атакующих примеров	18	6	6	-	-	-	12	6	-	6
Тема 12. Порождающие модели	18	6	6	-	-	-	12	6	-	6
Тема 13. Методы генерации вредоносных искажений	30	10	10	-	-	-	20	10	-	10
Тема 14. Оценка устойчивости моделей МО к внешним воздействиям	30	10	10	-	-	-	20	10	-	10
Тема 15. Повышение ро-	24	8	8	-	-	-	16	8	-	8

бастности моделей МО										
Тема 16. Прикладные библиотеки	24	8	8	-	-	-	16	8	-	8
17. Промежуточная аттестация – экзамен										
Итого	216	54	54				108	108		108

6. Фонд оценочных средств (ФОС, оценочные и методические материалы) для оценивания результатов обучения по дисциплине (модулю).

6.1. Типовые контрольные задания или иные материалы для проведения текущего контроля успеваемости, критерии и шкалы оценивания

Примерные задания для практических занятий.

ПСР ТК1. 2-адический анализ и Т-функции

Примерные варианты заданий:

1. Вывести тождества, связывающие продолжения на все пространство целых 2 адических чисел функций, представляющих основные арифметические и поразрядные логические команды процессора (+, NOT,AND,OR,XOR).
2. Доказать равномерную непрерывность вышеуказанных функций.
3. Найти частные производные (и производные по модулю 2^n) вышеуказанных функций во всех тех случаях, когда эти производные существуют.

ПСР ТК2. Генераторы псевдослучайных чисел на основе Т-функций

Примерные варианты заданий:

1. Найти длину периода инверсивного генератора с законом рекурсии $f(x) = 1 + x + \frac{p^2}{1 + px}$.
2. Показать, что функция inv (взятие обобщенного обратного) детерминирована, дифференцируема везде кроме 0: и найти общий вид ее производной i-го порядка в точке, отличной от 0.
3. Доказать, что функция $\frac{(x^p - x)^2}{p}$ является A-функцией, но не является B-функцией.

Примеры заданий для самостоятельной работы
Примеры домашних работ

Задание 1. Реализовать на языке программирования Python 3 указанные ниже алгоритмы и программы. Разрешается использовать только стандартные библиотеки, если в подзадаче не указано иное. Для оценки выполнения заданий будут использованы тесты, проверяющие результат выполнения программы на заданных входных данных. Задача считается выполненной только в том случае, если реализация пр ошла все тесты.

Задача 1.1.

Почти каждые четыре года в календарь добавляется дополнительный день — 29 февраля, и этот день называется високосным. В григорианском календаре для определения високосных лет используются три условия:

- Год можно разделить на 4 без остатка, это високосный год, если только:
- Год можно разделить на 100 без остатка, это НЕ високосный год, если только:
- Год тоже без остатка делится на 400. Тогда это високосный год.

Это означает, что в григорианском календаре 2000 и 2400 годы являются високосными, а 1800, 1900, 2100, 2200, 2300 и 2500 НЕ являются високосными годами.

Учитывая год, определить, является ли он високосным. Если это високосный год, верните значение True, в противном случае верните False.

Необходимо реализовать только функцию is_leap.

Формат ввода

year — год для тестирования.

Ограничения

1900 <= year <= 10^5

Выходной формат

Функция должна возвращать True или False.

Пример

is_leap(1990) == False

Задача 1.2.

Рассмотрим список (list = []). Вы можете выполнять следующие команды:

- insert i e: Вставить целое число `e` в позицию `i`.
- remove e: Удалить первое вхождение целого числа `e`.
- append e: Вставить целое число `e` в конце списка.
- sort: Сортировка списка.
- pop: извлечь последний элемент из списка.
- reverse: перевернуть список.

Инициализируйте пустой список и выполните набор команд, где каждая команда будет одного из 6 типов, перечисленных выше. Выполняйте каждую команду по порядку.

Пример

process_list(["append 1", "append 2", "insert 1 3"]) == [1, 3, 2]

Формат ввода

Функция process_list принимает список строк. Каждая строка содержит одну из описанных выше команд.

Ограничения

Элементы, добавляемые в список, должны быть целыми числами.

Выходной формат

Список целых чисел.

Задача 1.3.

Реализуйте функцию mutate_string, которая изменяет символ по заданному индексу, а затем возвращает измененную строку.

mutate_string имеет следующие параметры:

1. string: строка для изменения
2. position: индекс для вставки символа
3. char: символ для вставки

mutate_string возвращает измененную строку

Пример

mutate_string("abracadabra", 5, "X") == "abracXdabra"

Задача 1.4.

Имея целое число n, создайте список строк.

Для каждого целого числа i от 1 до n строка с индексом i должна содержать следующие представления числа i:

1. Десятичное
1. Восьмеричное
1. Шестнадцатеричное (с заглавной буквы)
1. Бинарное

Формат ввода

number — int, максимальное значение

Вывод

Для каждой строки четыре значения должны быть включены в порядке, указанном выше.
Каждое значение должно быть дополнено пробелом, чтобы соответствовать ширине двоичного значения number, и значения должны быть разделены одним пробелом.

Ограничения

1 <= n <= 99

Пример

```
print_formatted(17) == [
    "    1     1     1     1",
    "    2     2     2    10",
    "    3     3     3    11",
    "    4     4     4   100",
    "    5     5     5   101",
    "    6     6     6   110",
    "    7     7     7   111",
    "    8    10     8  1000",
    "    9    11     9  1001",
    "   10    12     A  1010",
    "   11    13     B  1011",
    "   12    14     C  1100",
```

```
    "   13    15     D  1101",
    "   14    16     E  1110",
    "   15    17     F  1111",
    "   16    20    10 10000",
    "   17    21    11 10001",
]
```

Подсказки

1. Существуют встроенные функции, такие как `bin(n)`, `hex(n)` и `oct(n)`. Запустите интерпретатор Python и проверьте, как они работают.
2. https://docs.python.org/3/library/string.html#formatstrings

Задача 1.5.

У вас есть непустой набор s, и вы должны выполнить на нем заданные команды.

Доступные команды — `pop`, `remove n` и `discard n`.
Используйте соответствующие методы структуры данных set (например, `your_set.pop()`). Выясните, что они делают и в чем разница.

Формат ввода

1. Список номеров для создания набора. Все элементы представляют собой неотрицательные целые числа, меньшие или равные 9.
2. Список строк. Каждая строка представляет собой команду pop, remove или discard, за которой следует соответствующее значение.

Выходной формат

Функция `process_set` должна возвращать сумму элементов набора в одной строке.

Пример

```питон
process_set(
```

```
    [1, 2, 3, 4, 5, 6, 7, 8, 9],
    [
        "pop",
        "remove 9",
        "discard 9",
        "discard 8",
        "remove 7",
        "pop",
        "discard 6",
        "remove 5",
        "pop",
        "discard 5",
    ]
) == 4
```

Задача 1.6.

Вам будет предоставлено два списка (A и B) строк. Для каждой строки `sb` из списка B проверьте, есть ли она в списке A или нет.
Создайте список позиций (индекс + 1) каждого вхождения `sb` в списке A. Если он не появляется, добавьте к списку -1.
Выведите список таких списков (длина равна длине списка B).

Пример

Список A содержит «a», «б», «a». Список B содержит «a», «c»

Первая строка в списке B, 'a', появляется на позициях 1 и 3 в списке A.
Второе слово, 'c', не появляется в группе A, поэтому список соответствующих позиций будет [-1].

```
lists_intersections(
    ['a', 'b', 'a'],    # list A
    ['a', 'c'],         # list B
) == [
    [1, 3],             # positions of the first string of list B
    [-1],               # positions of the second string of list B
```

]

Формат ввода

lists_intersections принимает два списка строк.

Ограничения

1 <= длина каждой строки в списке A или B <= 100

Выходной формат

Список списков целых чисел.

Пример 2

```
lists_intersections(
    ["a", "a", "b", "a", "b"]
    ["a", "b", "c"]
) == [
    [1, 2, 4],                 # 'a' appeared 3 times in positions 1, 2 and 4.
    [3, 5],                    # 'b' appeared 2 times in positions 3 and 5.
    [-1],                      # there is no `c` in list A => -1.
]
```

Подсказка: используйте `from collections import defaultdict`, чтобы импортировать структуру defaultdict.

Задача 1.7.

Вам дана строка `S`.
Ваша задача — выяснить, является ли `S` допустимым регулярным выражением или нет.

Пример

Функция is_valid_regex принимает строку и должна возвращать логическое значение.

```
is_valid_regex(r".*\+") == True
is_valid_regex(r".*+")  ==  False
```

Задача 1.8.

Прочитайте четыре числа `a`, `b`, `c` и `d`, и вычислите результат a^b + c^d.

Пример

```
big_calc(9, 29, 7, 27) == 47101944096086083692017432 32
```

Ограничения

1 <= a,b,c,d <= 1000

Примечание. Этот результат больше, чем 2^63 - 1. Следовательно, он не поместился бы в переменную типа long long int в C++.

Задание 2. Это задание является продолжением задания №1. Необходимо реализовать на языке программирования Python 3 указанные ниже алгоритмы и программы. Разрешается использовать только стандартные библиотеки, если в подзадаче не указано иное. Для оценки выполнения заданий будут использованы тесты, проверяющие результат выполнения программы на заданных входных данных, а также проверяющих время исполнения. Задача считается выполненной только в том случае, если реализация прошла все тесты и время исполнения для каждого тесты не превысило заданный порог.

Задача 2.1.

Реализовать генератор чисел Фибоначчи.

Пример работы

```
g = fib()
next(g) # == 0
next(g) # == 1
next(g) # == 1
next(g) # == 2
…
```

Задача 2.2.

Описание

Реализовать генератор, который принимает на вход конечное число списков целых чисел, а возвращает сначала все четные элементы, а потом все нечетные элементы из всех списков.

Пример работы

```
g = even_odd([1,2,3], [4,5,6])
next(g) # == 2
next(g) # == 4
next(g) # == 6
next(g) # == 1
next(g) # == 3
next(g) # == 5
next(g) # raises StopIteration
...
```

Примечание

Наивное решение может не пройти по времени. Задачу можно эффективно решить при помощи:
1. https://docs.python.org/3/library/itertools.html#itertools.tee
2. https://docs.python.org/3/library/itertools.html#itertools.chain

Задача 2.3.

Описание

Реализовать функцию `counter`, которая первым параметром получает некоторый класс, а остальные параметры применяет для создания экземпляра этого класса.

Функция должна возвращать 4 отсортированных списка:
1. имена методов класса
1. имена полей класса
1. имена методов, которые появились в экземпляре (т. е. в классе их не было, а при создании экземпляра они появились)
1. имена полей, которые появились в экземпляре (под «полями» имеются в виду не-callable() атрибуты).

Пример работы

```
class C:
    x, y, z = 1, 3, 5

    def X(self): return self.x
    def Y(self): return self.y

    def __init_(self, dx, dy, dz):
        self.x = dx
        self.Y = dy
        self.Z = dz

cm, cf, om, of = counter(C, 6, 7, 8)
# cm == [X, Y]
# cf == [x, y, z]
# om == []
# of == [Y Z]
```

Задача 2.4.

Описание

Написать класс C, экземпляры которого:

1. можно создать из чего угодно (в т. ч. из ничего)
2. можно индексировать по любому значению (возвращается объект, который использовался в качестве индекса)
3. позволяют выполнить присваивание и удаление элементов по индексу (эта операция не делает ничего)
4. содержат любое поле (возвращается имя этого поля)
5. позволяют присваивать и удалять поля (эта операция не делает ничего)
6. итерируемы (последовательность всегда пуста)
7. в виде строки представляются как "C"

Входные данные

```
M, N = C(), C(1,2,3,4)
print(M, N) # "C C"
M[13] = N.abc = 37
print(M[13], N.abc) # 13 abc
print(*list(C("ABCBA"))) # prints empty string
del M["QQ"], N[6:10], M[...], N._
print(M.adhd, N[-2]) # adhd -2
```

Задача 2.5.

Описание

Реализовать класс StringMinus, который бы полностью воспроизводил поведение встроенного класса str, но дополнительно поддержи вал бы операцию вычитания строк.

Вычитание работает по следующему правилу:
уменьшаемое просматривается посимвольно, если текущий символ присутствует в вычитаемом, то он однократно удаляется из обеих строк.

Пример работы

```
StringMinus("qwertyerty") - StringMinus("ttttr") == "qweyery" # True
```

Задача 2.6.

Описание

Написать параметрический декоратор `MyPy`, который принимает первым аргументом последовательность типов (`args_types`), а второй тип результата работы функции (`res_type`). Он должен бросать исключение TypeError при вызове функции со следующим сообщением:

- "Type of argument Номер is not Тип", если не совпадает тип позиционного параметра функции и соответствующий ему по порядку тип в args_types
- "Type of argument Имя is not Тип", если не совпадает тип именного параметра функции и соответствующий ему тип в args_types. Типы именованных параметров перечислены в конце args_types в порядке их описания в сигнатуре декорируемой функции.
- "Type of result is not Тип", если тип возвращённого функцией значения не совпадает с res_type
- "Function Функция must have Число arguments", если количество переданных функции параметров (включая переданные по умолчанию) не соответствует длине args_types

Сначала проверяются параметры в порядке описания в функции, затем вызывается функция, после чего проверяется результат.
Исключение возникает при первом несовпадении типа.

Примечание

Кратко про *args, **kwargs и декораторы: https://www.ritchieng.com/python/decorators-kwargs-args/

Параметрический декоратор отличается от обычного тем, что принимает аргументы и возвращает декоратор (то есть функцию).
Получается своего рода конструктор декораторов.

Пример параметрического декоратора:

```
def times(times):
    def inner(func):
        def newfunc(*args):
            return [func(*args) for i in range(times)]
        return newfunc
    return inner
```

```
@times(5)
def say_hi():
    return "hi"

print(*say_hi()) # print hi 5 times
```

Задание 3. В данном задании необходимо реализовать на языке Python 3 модуль для работы со скалярами. Вам будет предоставлен шаблон проекта, в котором уже реализована структура модуля, а также приложены тесты для данного задания. Разрешается использовать только те библиотеки, который уже используются в представленном шаблоне. Оценка за задание соответствует числу успешно выполняющихся те - стов. Вы можете самостоятельно проверить себя используя приложенные тесты, но их исправление запрещено. Исключение составляют части задания, в которых явно указано, что необходимо исправить тот или иной тест.

Задача 3.1.

Реализуйте следующий набор операторов:

operators.mul(*x*, *y*)
f(x,y)=x∗y

operators.id(*x*)
f(x)=x

operators.eq(*x*, *y*)
f(x)= 1.0 if x is equal to y else 0.0

operators.neg(*x*)
f(x)=−x

operators.add(*x*, *y*)
f(x,y)=x+y

operators.max(*x*, *y*)
f(x)= x if x is greater than y else y

operators.lt(*x*, *y*)
f(x)= 1.0 if x is less than y else 0.0

operators.sigmoid(*x*)
(See https://en.wikipedia.org/wiki/Sigmoid_function .)

Calculate as

$f(x) = \frac{1.0}{(1.0+e^{-x})}$ if x >=0 else $\frac{e^x}{(1.0+e^x)}$

for stability.

Parameters: **x** (*float*) -- input
Returns: sigmoid value
Return type: float

operators.relu(*x*)
f(x)= x if x is greater than 0, else 0

(See https://en.wikipedia.org/wiki/Rectifier_(neural_networks) .)

Parameters
x (*float*) -- input
Returns
relu value
Return type
float

operators.inv(*x*)
f(x)=1/x

operators.inv_back(*x*, *d*)
If f(x)=1/x compute d*f′(x)

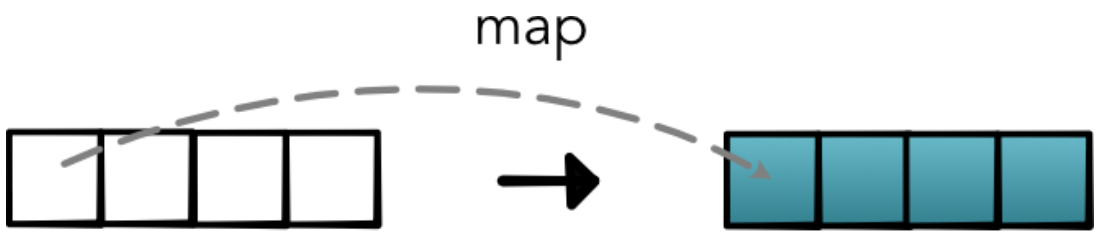
operators.relu_back(*x*, *d*)
If f=relu compute d*f′(x)

operators.log_back(*x*, *d*)
If f=log as above, compute d*f′(x)

Задача 3.2.

Реализуйте следующие функции:

operators.map(*fn*)
Higher-order map.



See https://en.wikipedia.org/wiki/Map_(higher-order_function)

Parameters

fn (*one-arg function*) -- Function from one value to one value.

Returns

A function that takes a list, applies fn to each element, and returns a new list

Return type

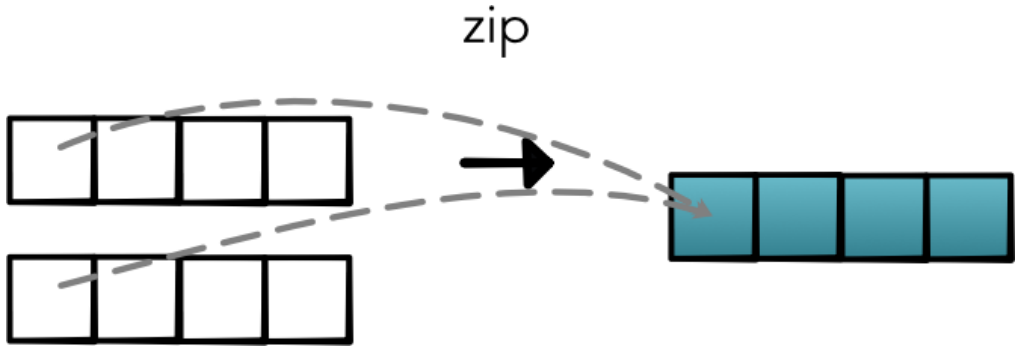
function

operators.negList(*ls*)

Use **map()** and **neg()** to negate each element in ls

operators.zipWith(*fn*)

Higher-order zipwith (or map2).



See https://en.wikipedia.org/wiki/Map_(higher-order_function)

Parameters

fn (*two-arg function*) -- combine two values

Returns

takes two equally sized lists ls1 and ls2, produce a new list by applying fn(x, y) on each pair of elements.

Return type

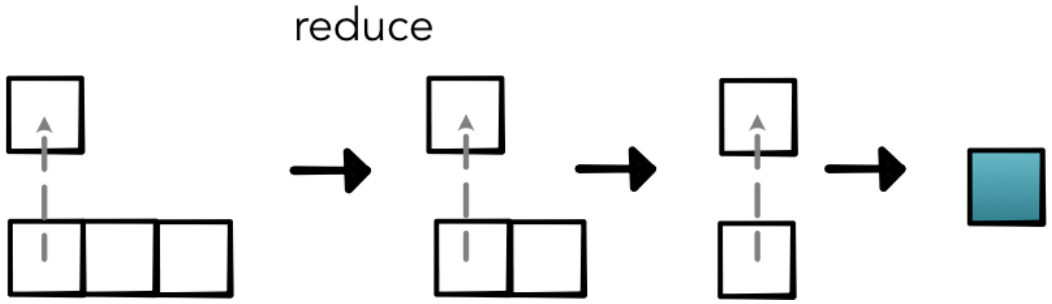
function

operators.addLists(*ls1*, *ls2*)

Add the elements of ls1 and ls2 using **zipWith()** and **add()**

operators.reduce(*fn*, *start*)

Higher-order reduce.



Parameters

- fn (*two-arg function*) -- combine two values
- start (*float*) -- start value x0

Returns

function that takes a list ls of elements x1…xn and computes the reduction fn(x3,fn(x2,fn(x1,x0)))

Return type

function

operators.sum(*ls*)

Sum up a list using **reduce()** and **add()**.

operators.prod(*ls*)

Product of a list using **reduce()** and **mul()**.

Задание 4. В данном задании необходимо реализовать на языке Python 3 модуль для автоматического дифференцирования. Вам будет предоставлен шаблон проекта, в котором уже реализована структура модуля, а также приложены тесты для данного задания. Разреша ется использовать только те библиотеки, который уже используются в представленном шаблоне. Оценка за задание соответствует числу успешно выполняющихся тестов. Вы можете самостоятельно проверить себя используя приложенные тесты, но их исправление запрещено. Искл ючение составляют части задания, в которых явно указано, что необходимо исправить тот или иной тест.

Задание 4.1.

Реализуйте функцию для численного дифференцирования функции, которая принимает на вход функцию, n аргументов этой функции, но - мер аргумента, по которому производить дифференцирование, а также значение параметра epsilon.

scalar.central_difference(*f, *vals, arg=0, epsilon=1e-06*)

Parameters:
- **f** -- arbitrary function from n-scalar args to one value
- ***vals** (*list of floats*) -- n-float values $x_0 \ldots x_{n-1}$
- **arg** (*int*) -- the number i of the arg to compute the derivative
- **epsilon** (*float*) -- a small constant

Returns: An approximation of $f_i'(x_0, \ldots, x_{n-1})$

Return type: float

Задание 4.2.

В представленном шаблоне есть класс Scalar. Необходимо реализовать недостающие forward и backward шаги для всех математических операций, которые есть в этом классе в виде методов:

scalar.Mul.forward(*ctx, a, b*)
scalar.Inv.forward(*ctx, a*)
scalar.Neg.forward(*ctx, a*)
scalar.Sigmoid.forward(*ctx, a*)
scalar.ReLU.forward(*ctx, a*)
scalar.Exp.forward(*ctx, a*)

scalar.LT.forward(*ctx, a, b*)
scalar.EQ.forward(*ctx, a, b*)fj

scalar.Mul.backward(*ctx, d_output*)
scalar.Inv.backward(*ctx, d_output*)
scalar.Neg.backward(*ctx, d_output*)
scalar.Sigmoid.backward(*ctx, d_output*)
scalar.ReLU.backward(*ctx, d_output*)
scalar.Exp.backward(*ctx, d_output*)

Также необходимо реализовать следующие операторы сравнения и функции:

Scalar.__lt__(*self, b*)
Scalar.__gt__(*self, b*)
Scalar.__sub__(*self, b*)
Scalar.__neg__(*self*)
Scalar.__add__(*self, b*)
Scalar.log(*self*)
Scalar.exp(*self*)
Scalar.sigmoid(*self*)
Scalar.relu(*self*)

Задание 4.3.

Реализуйте функцию для дифференцирования сложной функции (цепное правило, chain rule).

FunctionBase.chain_rule(*ctx, inputs, d_output*)

Parameters:
- **ctx** (`Context`) -- The context from running forward
- **inputs** (*list of args*) -- The args that were passed to `FunctionBase.apply()` (e.g. x, y)
- **d_output** (*number*) -- The *d_output* value in the chain rule.

Returns: A list of non-constant variables with their derivatives (see *is_constant* to remove unneeded variables)

Return type: list of (*Variable*, number)

Задание 4.4.
Реализуйте метод обратного распространения ошибки. Он не должен ничего возвращать — результат должен суммироваться в атрибуте *accumulate_derivative* для каждого листа (переменной) в графе вычисления.

backpropagate(*variable, deriv*)

Parameters:
- **variable** (`Variable`) -- The right-most variable
- **deriv** (*number*) -- Its derivative that we want to propagate backward to the leaves.

Задание 5. В данном задании необходимо реализовать на языке Python 3 модуль для работы с тензорами. Вам будет предоставлен шаблон проекта, в котором уже реализована структура модуля, а также приложены тесты для данного задания. Разрешается использовать только те библиотеки, который уже используются в представленном шаблоне. Оценка за задание соответствует числу успешно выполняющихся те - стов. Вы можете самостоятельно проверить себя используя приложенные тесты, но их исправление запрещено. Исключение составляют ча- сти задания, в которых явно указано, что необходимо исправить тот или иной тест.

Задача 5.1. Реализуйте функции индексации тензоров, а также функцию для изменения размерности.

index_to_position(*index, strides*)
Converts a multidimensional tensor index into a single-dimensional position in storage based on strides.

Parameters:
- **index** (*array-like*) -- index tuple of ints
- **strides** (*array-like*) -- tensor strides

Returns: position in storage

Return type: int

index_to_position(*index*, *strides*)

Converts a multidimensional tensor index into a single-dimensional position in storage based on strides.

Parameters:
- **ordinal** (*int*) -- ordinal position to convert.
- **shape** (*tuple*) -- tensor shape.
- **out_index** (*array*) -- the index corresponding to position.

Returns: Fills in *out_index*.

Return type: None

TensorData.permute(*self*, **order*)

Permute the dimensions of the tensor.

Parameters: **order** (*list*) -- a permutation of the dimensions

Returns: a new TensorData with the same storage and a new dimension order.

Return type: `TensorData`

Задача 5.2.

Реализуйте функции для бродкастинга тензоров:

shape_broadcast(*shape1*, *shape2*)

Broadcast two shapes to create a new union shape.

Parameters:
- **shape1** (*tuple*) -- first shape
- **shape2** (*tuple*) -- second shape

Returns: broadcasted shape

Return type: tuple

Raises: **IndexingError** -- if cannot broadcast

broadcast_index(*big_index, big_shape, shape, out_index*)

Convert a big_index into big_shape to a smaller out_index into shape following broadcasting rules. In this case it may be larger or with more dimensions than the shape given. Additional dimensions may need to be mapped to 0 or removed.

Parameters:
- **big_index** (*array-like*) -- multidimensional index of bigger tensor
- **big_shape** (*array-like*) -- tensor shape of bigger tensor
- **shape** (*array-like*) -- tensor shape of smaller tensor
- **out_index** (*array-like*) -- multidimensional index of smaller tensor

Returns: Fills in *out_index*.

Return type: None

Задание 5.3.

Реализуйте функции map, zip и reduce для тензоров.

tensor_map(*fn*)

Parameters

- fn -- function mappings floats-to-floats to apply.
- out (*array*) -- storage for out tensor.
- out_shape (*array*) -- shape for out tensor.
- out_strides (*array*) -- strides for out tensor.
- out_size (*array*) -- size for out tensor.
- in_storage (*array*) -- storage for in tensor.
- in_shape (*array*) -- shape for in tensor.
- in_strides (*array*) -- strides for in tensor.

Returns

Fills in out

Return type

None

tensor_zip(*fn*)

Parameters

- fn -- function mappings two floats to float to apply.
- out (*array*) -- storage for out tensor.
- out_shape (*array*) -- shape for out tensor.
- out_strides (*array*) -- strides for out tensor.
- out_size (*array*) -- size for out tensor.
- a_storage (*array*) -- storage for a tensor.
- a_shape (*array*) -- shape for a tensor.
- a_strides (*array*) -- strides for a tensor.
- b_storage (*array*) -- storage for b tensor.
- b_shape (*array*) -- shape for b tensor.
- b_strides (*array*) -- strides for b tensor.

Returns

Fills in out

Return type

None

tensor_reduce(*fn*)

Parameters

- fn -- reduction function maps two floats to float.
- out (*array*) -- storage for out tensor.
- out_shape (*array*) -- shape for out tensor.
- out_strides (*array*) -- strides for out tensor.
- out_size (*array*) -- size for out tensor.
- a_storage (*array*) -- storage for a tensor.
- a_shape (*array*) -- shape for a tensor.
- a_strides (*array*) -- strides for a tensor.
- reduce_dim (*int*) -- dimension to reduce out

Returns

Fills in out

Return type

None

Задание 5.4.

Реализуйте в модуле TensorFunctions операции для тензоров (forward и backward шаги).

TensorFunctions.Mul.forward(*ctx, a, b*)
TensorFunctions.Sigmoid.forward(*ctx, a*)
TensorFunctions.ReLU.forward(*ctx, a*)
TensorFunctions.Log.forward(*ctx, a*)
TensorFunctions.Exp.forward(*ctx, a*)
TensorFunctions.LT.forward(*ctx, a, b*)
TensorFunctions.EQ.forward(*ctx, a, b*)
TensorFunctions.Permute.forward(*ctx, a, order*)

TensorFunctions.Mul.backward(*ctx, grad_output*)
TensorFunctions.Sigmoid.backward(*ctx, grad_output*)
TensorFunctions.ReLU.backward(*ctx, grad_output*)

TensorFunctions.Log.backward(*ctx, grad_output*)
TensorFunctions.Exp.backward(*ctx, grad_output*)
TensorFunctions.LT.backward(*ctx, grad_output*)
TensorFunctions.EQ.backward(*ctx, grad_output*)
TensorFunctions.Permute.backward(*ctx, grad_output*)

Задача 6. В рамках данного задания вам предстоит написать нейросеть для бинарной классификации изображений — настоящих и поддельных. Это задание выполняется на платформе Kaggle в виде соревнования. Пригласительная ссылка на указанный ресурс будет выслан а каждому студенту. Перейдя по ней, вы получите доступ к набору данных для обучения. Во вкладке overview можно найти дополнительные све - дения. Также будет приложено базовое решение данной задачи. Вы можете использовать базовое решение как основу для своего, улу чшая его и повышая качество распознавания. Для обучения моделей можно воспользоваться либо своими собственными вычислительными рес урсами, либо ресурсами, которые бесплатно предоставляет платформа Kaggle (для этого необходимо дополнительно подтвердить свою у четную запись в настройках). Критерий оценивания — значение метрики ROC AUC на тестовом наборе данных. Кроме того, будет проведена проверка каждого решения на предмет плагиата. В случае обнаружения значительного заимствования чужого кода, задание засчита но не будет.

6.2. Типовые контрольные задания или иные материалы для проведения промежуточной аттестации по дисциплине, критерии и шкалы оценивания

Список вопросов для экзамена.

1. Вектор, скалярное и векторное произведение, линейная оболочка.
2. Матрица, определитель и след матрицы, норма.
3. Алгоритмы умножение матриц, нахождение определителя и обратной матрицы, спектральное и сингулярное разложений матриц, метод главных компонент.
4. Дискретные и непрерывные случайные величины, частота распределения, функции вероятности и плотности вероятности.
5. Маргинальное распределение вероятности, условная вероятность, цепное правило, формула Байеса.
6. Мода, медиана, математическое ожидание, дисперсия, ковариация, корреляция, матрица ковариации.
7. Распределения Бернулли, категориальное, нормальное, Лапласа, Дирака, эмпирическое, комбинации распределений.
8. Собственная информация, понятие ната, энтропия Шеннона, расхождение Кульбака-Лейблера, перекрестная энтропия.
9. Производные разных порядков, поиск максимумов и минимумов функции, производные по направлению, градиент, матрица Гессе, функция Лагранжа, разложение функции в ряд Тейлора.
10. Методы градиентного спуска, Каруша-Куна-Таккера, наименьших квадратов.

11. Алгоритмы обучения. Переобучение и недообучение. Гипперпараметры и контрольные наборы. Оценки, смещения, дисперсия.
12. Оценка максимального правдоподобия. Байсовская статистика.
13. Алгоритмы обучения с учителем. Алгоритмы обучения без учителя. Стохастический градиентный спуск.
14. ИНС прямого распространения, сверточные сети, рекуррентные сети, автоэнкодеры.
15. ГНС для обработки изображений, звука и видеоряда.
16. ГНС для обработки текстовых данных.
17. Базовые принципы создания атакующих примеров.
18. Порождающие модели.
19. Методы генерации вредоносных искажений.
20. Оценка устойчивости моделей МО к внешним воздействиям.
21. Повышение робастности моделей МО.

ШКАЛА И КРИТЕРИИ ОЦЕНИВАНИЯ результатов обучения (РО) по дисциплине

Оценка / виды оценочных средств	2 (не зачтено)	3 (зачтено)	4 (зачтено)	5 (зачтено)
Знания *(виды оценочных средств: опрос, тесты)*	Отсутствие знаний	Фрагментарные знания	Общие, но не структурированные знания	Сформированные систематические знания
Умения *(виды оценочных средств: практические задания)*	Отсутствие умений	В целом успешное, но не систематическое умение	В целом успешное, но содержащее отдельные пробелы умение (допускает неточности непринципиального характера)	Успешное и систематическое умение
Навыки (владения, опыт деятельности) *(виды оценочных средств: выполнение и защита курсовой ра-*	Отсутствие навыков (владений, опыта)	Наличие отдельных навыков (наличие фрагментарного опыта)	В целом, сформированные навыки (владения), но используемые не в активной форме	Сформированные навыки (владения), применяемые при решении задач

	боты, отчет по практике, отчет по НИР и т.п.)				

Методические материалы для проведения процедур оценивания результатов обучения

Особенности организации процесса обучения

Для эффективного освоения курса рекомендуется перед каждым занятие привести в порядок конспекты лекций. После каждого занятия рекомендуется найти и прочитать дополнительную литературу по теме лекции и прочитать свои конспекты.

Система контроля и оценивания

За каждую домашнюю выставляются баллы (максимум 40 баллов). Пусть М – максимальное число баллов, которое может набрать студент. В конце семестра баллы конвертируются в оценку О1 следующим образом:

меньше М/2 баллов: О1=2;

больше или равно М/2 баллов, но меньше 2М/3: О1=3;

больше или равно 2М/3 баллов, но меньше 5М/6: О1=4;

больше или равно 5М/6 баллов: О1=5.

На экзамене оценка О1 является стартовой. Окончательная оценка определяется исходя из оценки устного ответа студента, при этом она не может отличаться от стартовой оценки более чем на 1 балл.

Структура и график контрольных мероприятий

Устная сдача домашних заданий в конце каждой недели, устный экзамен в конце каждого семестра.

Основная литература:

Основная литература

1. Бенджио Иошуа, Гудфеллоу Ян, Курвилль Аарон, «Глубокое обучение», ДМК Пресс, 2017.
2. Katy Warr, « Strengthening Deep Neural Networks», O'Reilly Media, 2019.
3. Jerry Zheng Li, «Principled Approaches to Robust Machine Learning and Beyond», MIT, 2018.

Дополнительная литература:

1. Stephen Boyd, Lieven Vandenberghe, «Introduction to Applied Linear Algebra», Cambridge University Press, 2018.
2. Garrett Thomas, «Mathematics for Machine Learning», University of California, Berkeley, 2018.
3. PyTorch, https://pytorch.org.
4. TensorFlow, https://www.tensorflow.org.

7.2. Перечень лицензионного программного обеспечения, в том числе отечественного производстваПри реализации дисциплины может быть использовано следующее программное обеспечение:

1. Программное обеспечение для подготовки слайдов лекций MS PowerPoint
2. Программное обеспечение для создания и просмотра pdf-документов Adobe Reader
3. Издательская система LaTeX.

7.3. Перечень профессиональных баз данных и информационных справочных систем

1. http://www.edu.ru – портал Министерства образования и науки РФ
2. http://www.ict.edu.ru – система федеральных образовательных порталов «ИКТ в образовании»
3. http://www.openet.ru - Российский портал открытого образования
4. http://www.mon.gov.ru - Министерство образования и науки Российской Федерации
5. http://www.fasi.gov.ru - Федеральное агентство по науке и инновациям

7.4. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

1. Math-Net.Ru [Электронный ресурс] : общероссийский математический портал / Математический институт им. В. А. Стеклова РАН ; Российская академия наук, Отделение математических наук. - М. : [б. и.], 2010. - Загл. с титул. экрана. - Б. ц.
URL: http://www.mathnet.ru
2. Университетская библиотека Online [Электронный ресурс] : электронная библиотечная система / ООО "Директ-Медиа" . - М. : [б. и.], 2001. - Загл. с титул. экрана. - Б. ц. URL: www.biblioclub.ru
3. Универсальные базы данных East View [Электронный ресурс] : информационный ресурс / East View Information Services. - М. : [б. и.], 2012. - Загл. с титул. экрана. - Б. ц.

URL: www.ebiblioteka.ru

4. Научная электронная библиотека eLIBRARY.RU [Электронный ресурс] : информационный портал / ООО "РУНЭБ" ; Санкт-Петербургский государственный университет. - М. : [б. и.], 2005. - Загл. с титул. экрана. - Б. ц.
URL: www.eLibrary.ru

7.5. Описание материально-технического обеспечения.

Факультет ВМК, ответственный за реализацию данной Программы, располагает соответствующей материально -технической базой, включая современную вычислительную технику, объединенную в локальную вычислительную сеть, имеющую выход в Интернет. Используются специализированные компьютерные классы, оснащенные современным оборудованием. Материальная база факультета соответствует действующим санитарно-техническим нормам и обеспечивает проведение всех видов занятий (лабораторной, практической, дисциплинарной и междисциплинарной подготовки) и научно-исследовательской работы обучающихся, предусмотренных учебным планом.

8. Соответствие результатов обучения по данному элементу ОПОП результатам освоения ОПОП указано в Общей характеристике ОПОП.

9. Разработчик (разработчики) программы.

Строева Екатерина Николаевна, Малоян Нарек Гагикович, Саада Даниель Фирасович, Ильюшин Евгений Альбинович, Намиот Дмитрий Евгеньевич.

10. Язык преподавания - русский.