МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению практических работ
по дисциплине «Основы нейронных сетей»
для студентов направления подготовки
11.03.02 Инфокоммуникационные технологии и системы связи
Направленность (профиль)
«Интеллектуальная обработка данных в инфокоммуникационных системах и сетях»

Ставрополь 2026

Содержание

ВВЕДЕНИЕ

1. Цель и задачи освоения дисциплины (модуля)

Цель дисциплины – формирование, согласно образовательной программы, компетенции ПК-1 обучающегося.

Для достижения цели освоения дисциплины «Основы нейронных сетей» в процессе обучения решаются следующие задачи: ознакомление с биологическими предпосылками возникновения искусственных нейронных сетей и формированием математической модели искусственного нейрона; изучение принципов построения однослойных и многослойных нейронных сетей, включая модель персептрона и её ограничения; освоение математических основ обучения нейронных сетей как задачи оптимизации, включая функции потерь для задач регрессии и классификации; изучение методов градиентной оптимизации, включая стохастический и мини-батч градиентный спуск; овладение современными методами ускорения обучения нейронных сетей, включая алгоритмы Momentum, AdaGrad, RMSprop, Adadelta, Adam и их модификации; понимание факторов, влияющих на сходимость и скорость обучения нейронных сетей; освоение практических навыков реализации и обучения нейронных сетей на языке Python с использованием библиотеки PyTorch; развитие умений применять нейронные сети для решения задач классификации и регрессии, возникающих при анализе данных и сигналов в инфокоммуникационных системах; формирование навыков проведения вычислительных экспериментов, анализа результатов обучения моделей и интерпретации полученных решений.

2. Место дисциплины (модуля) в структуре образовательной программы

Дисциплина относится к факультативным дисциплинам.

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов
ПК-1. Способен к развитию коммутационных подсистем и сетевых платформ, сетей передачи данных, транспортных сетей и сетей радиодоступа, спутниковых систем связи	ИД-2$_{ПК-1}$ анализирует статистические параметры трафика, вырабатывает решения по оперативному переконфигурированию сети, изменению параметров коммутационной подсистемы, сетевых платформ, по применению оборудования новых технологий; изменению и корректировке параметров коммутационной подсистемы, транспортных сетей и сетей передачи данных, по организации новых и расширению имеющихся направлений связи.	Проведение анализа статистических параметров трафика; выработка решений по оперативному переконфигурированию сети; изменение параметров коммутационной подсистемы и сетевых платформ; применение оборудования новых технологий; изменение и корректировка параметров коммутационной подсистемы, транспортных сетей и сетей передачи данных;

		организация новых и расширение имеющихся направлений связи.
	ИД-3$_{ПК-1}$. анализирует статистику основных показателей эффективности радиосистем и систем передачи данных, разрабатывает мероприятия по их поддержанию на требуемом уровне, выполняет расчет пропускной способности сетей телекоммуникаций.	Проведение анализа статистики основных показателей эффективности радиосистем и систем передачи данных; разработка мероприятий по поддержанию показателей эффективности радиосистем и систем передачи данных на требуемом уровне; выполнение расчета пропускной способности сетей телекоммуникаций.
	ИД-4$_{ПК-1}$ разрабатывает схемы организации связи и интеграции новых сетевых элементов, осуществляет построение и расширение коммутационной подсистемы и сетевых платформ; выполняет работы на коммутационном оборудовании, развертывает оборудование сервисных платформ, оборудование новых технологий на сети, выполняет планы по расширению существующего оборудования сетевых платформ и новых технологий.	Разработка схем организации связи и интеграции новых сетевых элементов; осуществление построения и расширения коммутационной подсистемы и сетевых платформ; выполнение работ на коммутационном оборудовании; развертывание оборудования сервисных платформ и оборудования новых технологий на сети; выполнение планов по расширению существующего оборудования сетевых платформ и новых технологий.

4. Объем учебной дисциплины (модуля) и формы контроля *

Объем занятий: всего: 3 з.е. 108 ак.ч.	ОФО, В акад. часах
Контактная работа:	24.00/18.00
Лекции/из них практическая подготовка	8.00/0
Лабораторных работ/из них практическая подготовка	-
Практических занятий/из них практическая подготовка	16.00/0
Самостоятельная работа	84.00
Формы контроля	
Экзамен	нет
Зачет	нет

Зачет с оценкой	да
Курсовая работа	нет

5. Важность практической работы студентов

Практическая работа является важным элементом образовательного процесса по дисциплине «Основы нейронных сетей», поскольку позволяет студентам закрепить теоретические знания и получить практический опыт разработки и обучения моделей нейронных сетей. Выполнение практических заданий обеспечивает переход от изучения математических принципов и архитектур нейронных сетей к их непосредственной реализации и применению для решения задач анализа данных.

В ходе практических занятий студенты осваивают основные этапы разработки нейросетевых моделей: подготовку данных, построение архитектуры сети, настройку параметров обучения, выбор функций потерь и методов оптимизации, а также оценку качества полученных моделей. Практическая работа выполняется с использованием языка программирования Python и библиотеки PyTorch, что позволяет познакомиться с современными инструментами разработки систем глубокого обучения и получить навыки их применения на практике.

Практические задания включают реализацию искусственного нейрона и персептрона, построение многослойного персептрона и его применение для распознавания изображений набора MNIST, работу с механизмами подготовки данных, а также разработку и обучение более сложных архитектур, таких как сверточные и рекуррентные нейронные сети. Это позволяет студентам увидеть особенности различных архитектур и понять области их применения.

Выполнение практических работ способствует развитию навыков программирования, анализа данных и экспериментального исследования поведения моделей машинного обучения. Студенты учатся интерпретировать результаты обучения нейронных сетей, анализировать влияние параметров модели на качество решений и принимать обоснованные технические решения при разработке интеллектуальных систем.

Таким образом, практические занятия обеспечивают тесную связь теоретических основ нейронных сетей с их практическим применением и формируют у студентов устойчивые навыки разработки и использования моделей глубокого обучения при решении задач анализа данных в профессиональной деятельности.

ПРАКТИЧЕСКАЯ РАБОТА №1

Тема: Реализация искусственного нейрона и персептрона в PyTorch

Цель работы

Овладение навыками программной реализации математической модели искусственного нейрона и однослойной нейронной сети (персептрона) с использованием библиотеки PyTorch. Освоение принципов вычисления взвешенной суммы входных сигналов, применения функции активации и обучения модели для решения задачи бинарной классификации.

Теоретическое обоснование

Искусственные нейронные сети представляют собой один из ключевых инструментов современного машинного обучения и анализа данных. Основой любой нейронной сети является искусственный нейрон — математическая модель, имитирующая некоторые свойства биологических нейронов.

Модель искусственного нейрона принимает набор входных сигналов, каждому из которых соответствует весовой коэффициент. Полученное значение передаётся в функцию активации: $y = f(z)$

Функция активации вводит нелинейность в модель и позволяет нейронной сети аппроксимировать сложные зависимости между входными и выходными данными. В простейших моделях могут использоваться сигмоидальная функция, гиперболический тангенс или функция ReLU.

Персептрон представляет собой одну из первых моделей нейронных сетей и является однослойной нейронной сетью, используемой для решения задач бинарной классификации. В классической постановке персептрон формирует линейную границу разделения между двумя классами объектов. Решение принимается на основе знака линейной комбинации входных признаков.

В современных системах глубокого обучения реализация нейронных сетей осуществляется с использованием специализированных библиотек. Одной из наиболее распространённых является PyTorch — фреймворк для разработки и обучения моделей глубокого обучения на языке Python. PyTorch предоставляет удобные средства для работы с многомерными массивами данных (тензорами), автоматического вычисления градиентов и реализации алгоритмов оптимизации.

Выполнение данной работы позволяет на практике реализовать математическую модель нейрона, понять принципы вычисления выходного сигнала, а также познакомиться с базовыми механизмами обучения нейронной сети.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- NumPy
- Matplotlib (для визуализации результатов)

Методические указания по выполнению работы

В ходе выполнения работы студент должен последовательно реализовать модель искусственного нейрона и однослойной нейронной сети для решения задачи бинарной классификации.

На первом этапе необходимо реализовать математическую модель искусственного нейрона, вычисляющего взвешенную сумму входных признаков и применяющего функцию активации. Для этого используются тензоры PyTorch и базовые операции линейной алгебры.

На втором этапе реализуется однослойная нейронная сеть (персептрон). Для модели задаются веса и параметр смещения, после чего формируется функция, вычисляющая выход сети. Далее необходимо реализовать процесс обучения модели на основе минимизации функции потерь с использованием одного из методов оптимизации.

В процессе выполнения работы рекомендуется визуализировать результаты обучения и проанализировать влияние параметров модели на точность классификации.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Реализовать модель искусственного нейрона, вычисляющего взвешенную сумму входных признаков и применяющего функцию активации.
2. Реализовать однослойную нейронную сеть (персептрон) для решения задачи бинарной классификации.
3. Сгенерировать простой набор данных из двух классов и обучить модель персептрона для разделения этих классов.
4. Исследовать влияние функции активации на результаты классификации.
5. Визуализировать обученную модель и границу разделения классов на плоскости.

Контрольные вопросы

1. Что представляет собой искусственный нейрон и какие элементы входят в его математическую модель.
2. Как вычисляется выход искусственного нейрона.
3. Какую роль выполняют весовые коэффициенты и параметр смещения.
4. Что такое функция активации и зачем она используется.
5. Какие функции активации применяются в нейронных сетях.
6. Что представляет собой персептрон.
7. Какие задачи может решать однослойный персептрон.
8. Как формируется граница разделения классов в модели персептрона.
9. Какие возможности предоставляет библиотека PyTorch для реализации нейронных сетей.
10. В чём заключается процесс обучения нейронной сети.

ПРАКТИЧЕСКАЯ РАБОТА №2

Тема: Реализация многослойного персептрона с использованием `torch.nn.Module`

Цель работы

Овладение навыками разработки и обучения многослойной нейронной сети (многослойного персептрона) с использованием библиотеки PyTorch. Изучение принципов построения архитектуры нейронной сети, использования функций активации, реализации модели через класс `torch.nn.Module`, а также освоение базового цикла обучения нейронной сети.

Теоретическое обоснование

Многослойный персептрон (Multilayer Perceptron, MLP) является одной из базовых архитектур нейронных сетей и представляет собой сеть, состоящую из нескольких последовательно соединённых слоёв нейронов. В отличие от однослойного персептрона, многослойный персептрон способен аппроксимировать сложные нелинейные зависимости между входными и выходными данными.

Каждый слой нейронной сети выполняет линейное преобразование входного вектора с последующим применением нелинейной функции активации. Для входного вектора признаков (x) выход слоя может быть представлен в виде

$$z = Wx + b$$

где
W — матрица весов,
b — вектор смещения,
z — линейное преобразование входных данных.

После этого применяется функция активации $a = f(z)$ где ($f()$) — нелинейная функция активации (например, ReLU, sigmoid или tanh). Наличие нелинейных функций активации позволяет сети моделировать сложные зависимости между признаками.

В процессе обучения параметры сети (веса и смещения) изменяются таким образом, чтобы минимизировать функцию потерь. Для этого используется алгоритм обратного распространения ошибки (backpropagation), который вычисляет градиенты функции потерь по параметрам модели. На основе вычисленных градиентов оптимизатор выполняет обновление параметров.

Библиотека PyTorch предоставляет удобные средства для построения и обучения нейронных сетей. Основой для реализации моделей является класс `torch.nn.Module`, который позволяет описывать архитектуру нейронной сети как объект, содержащий слои и функцию прямого распространения (`forward`).

Использование `nn.Module` обеспечивает удобное управление параметрами модели, автоматическое вычисление градиентов и интеграцию с алгоритмами оптимизации, реализованными в модуле `torch.optim`.

В рамках данной работы студенты реализуют многослойный персептрон, зададут его архитектуру с использованием полносвязных слоёв (`nn.Linear`) и функций активации, а затем обучат модель для решения задачи классификации.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- NumPy
- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы студент должен реализовать многослойный персептрон и обучить его на простом наборе данных.

На первом этапе необходимо создать класс модели, наследующий `torch.nn.Module`. Внутри класса следует определить архитектуру сети, включающую несколько полносвязных слоёв (`nn.Linear`) и функции активации.

Далее необходимо реализовать функцию прямого распространения (`forward`), которая определяет порядок прохождения входных данных через слои сети.

После создания модели требуется задать функцию потерь и алгоритм оптимизации. Затем реализуется цикл обучения, включающий следующие этапы:

- прямое распространение сигнала через сеть;
- вычисление функции потерь;
- вычисление градиентов методом обратного распространения ошибки;
- обновление параметров модели с использованием оптимизатора.

В процессе выполнения работы рекомендуется наблюдать за изменением значения функции потерь и анализировать поведение модели в ходе обучения.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Реализовать класс многослойного персептрона на основе `torch.nn.Module`, содержащий несколько полносвязных слоёв.
2. Добавить нелинейные функции активации между слоями сети.
3. Сгенерировать простой набор данных для задачи классификации и обучить модель многослойного персептрона.
4. Реализовать цикл обучения модели с использованием функции потерь и оптимизатора.
5. Проанализировать динамику изменения функции потерь в процессе обучения.

Контрольные вопросы

1. Что представляет собой многослойный персептрон.
2. Чем многослойный персептрон отличается от однослойного персептрона.
3. Какую роль выполняют функции активации в нейронной сети.
4. Для чего используется класс `torch.nn.Module`.
5. Что представляет собой функция `forward` в модели PyTorch.
6. Какие типы слоёв используются при построении многослойного персептрона.
7. Что такое алгоритм обратного распространения ошибки.
8. Какие компоненты входят в цикл обучения нейронной сети.
9. Для чего используются оптимизаторы при обучении моделей.
10. Какие преимущества предоставляет использование библиотеки PyTorch для разработки нейронных сетей.

ПРАКТИЧЕСКАЯ РАБОТА №3

Тема: Классификация изображений набора MNIST с использованием многослойного персептрона

Цель работы

Овладение навыками разработки и обучения модели многослойного персептрона для решения задачи классификации изображений. Изучение процесса подготовки данных, построения архитектуры нейронной сети и обучения модели с использованием библиотеки PyTorch на примере набора рукописных цифр MNIST.

Теоретическое обоснование

Задачи классификации изображений являются одной из наиболее распространённых областей применения нейронных сетей. В рамках таких задач модель должна определить принадлежность изображения к одному из заданных классов на основе анализа пиксельных значений.

Набор данных MNIST (Modified National Institute of Standards and Technology) является классическим набором данных, широко используемым для обучения и тестирования алгоритмов машинного обучения. Он содержит изображения рукописных цифр от 0 до 9 размером (28 \times 28) пикселей. Всего в наборе присутствует 70 000 изображений, из которых 60 000 используются для обучения модели, а 10 000 — для тестирования.

Каждое изображение можно представить в виде матрицы яркостей пикселей. Для использования в многослойном персептроне изображение обычно преобразуется в одномерный вектор:

$$x \in R^{784}$$

где $(784 = 28 \times 28)$ — общее число пикселей изображения.

Многослойный персептрон выполняет последовательность линейных преобразований и нелинейных функций активации:

$$a^{(l)} = f\left(W^{(l)}a^{(l-1)} + b^{(l)}\right)$$

где
$a^{(l)}$ — активации l-го слоя,
$W^{(l)}$ — матрица весов,
$b^{(l)}$ — вектор смещений,
$f(\cdot)$ — функция активации.

На выходном слое сети используется функция Softmax, позволяющая преобразовать выходы сети в вероятности принадлежности изображения к каждому из десяти классов:

$$P(y = i) = \frac{e^{z_i}}{\sum_{j=1}^{10} e^{z_j}}$$

Для обучения модели используется функция потерь кросс-энтропии, измеряющая различие между предсказанными вероятностями и истинными метками классов.

Библиотека PyTorch предоставляет готовые инструменты для работы с набором MNIST, включая загрузку данных, их преобразование в тензоры и организацию обучения с использованием классов `Dataset` и `DataLoader`.

Выполнение данной работы позволяет на практике реализовать полный цикл разработки нейронной сети: подготовку данных, построение модели, обучение и оценку качества классификации.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- torchvision
- NumPy

- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы необходимо реализовать модель многослойного персептрона и обучить её для классификации изображений набора MNIST.

На первом этапе следует загрузить набор данных MNIST с использованием средств библиотеки `torchvision.datasets`. Изображения необходимо преобразовать в тензоры и нормализовать.

Далее следует подготовить загрузчики данных (`DataLoader`) для обучающей и тестовой выборок, позволяющие передавать данные в модель небольшими пакетами (батчами).

После подготовки данных необходимо реализовать архитектуру многослойного персептрона с использованием `torch.nn.Module`. Входной слой должен принимать вектор размерности 784, далее используются один или несколько скрытых слоёв с функцией активации ReLU, а выходной слой содержит 10 нейронов — по числу классов.

Следующим этапом является определение функции потерь (кросс-энтропия) и оптимизатора. Затем реализуется цикл обучения, включающий прямое распространение, вычисление функции потерь, обратное распространение ошибки и обновление параметров модели.

После завершения обучения необходимо оценить точность модели на тестовой выборке и проанализировать результаты классификации.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Загрузить и визуализировать несколько изображений набора MNIST.
2. Подготовить обучающую и тестовую выборки с использованием `DataLoader`.
3. Реализовать модель многослойного персептрона для классификации изображений.
4. Обучить модель на обучающей выборке и отследить изменение функции потерь в процессе обучения.
5. Оценить точность модели на тестовой выборке и проанализировать ошибки классификации.

Контрольные вопросы

1. Что представляет собой набор данных MNIST.
2. Какова размерность изображений в наборе MNIST.
3. Почему изображения преобразуются в одномерный вектор при использовании многослойного персептрона.
4. Какую роль выполняет функция Softmax на выходном слое сети.
5. Какая функция потерь используется в задачах многоклассовой классификации.
6. Для чего используются загрузчики данных (`DataLoader`).
7. Что такое батч при обучении нейронной сети.
8. Какие этапы включает цикл обучения нейронной сети.
9. Как вычисляется точность модели классификации.
10. Какие преимущества предоставляет использование библиотеки PyTorch при разработке нейронных сетей.

ПРАКТИЧЕСКАЯ РАБОТА №4

Тема: Подготовка данных и организация процесса обучения нейронной сети с использованием `Dataset` и `DataLoader`

Цель работы

Овладение навыками подготовки данных для обучения нейронных сетей и организации процесса обучения моделей с использованием механизмов `Dataset` и `DataLoader` библиотеки PyTorch. Изучение принципов структурирования данных, формирования обучающих выборок и эффективной передачи данных в модель в процессе обучения.

Теоретическое обоснование

Качество обучения нейронной сети во многом определяется правильной подготовкой данных. Перед началом обучения данные необходимо структурировать, преобразовать в удобный для обработки формат и организовать их эффективную подачу в модель.

В задачах машинного обучения исходные данные обычно разделяются на несколько частей: обучающую выборку, используемую для обучения модели, и тестовую (или валидационную) выборку, применяемую для оценки качества работы модели. Такое разделение позволяет определить, насколько хорошо модель обобщает знания на новых данных.

При обучении нейронных сетей данные обычно передаются в модель не по одному объекту, а небольшими пакетами — батчами. Такой подход повышает эффективность вычислений и делает обучение более устойчивым.

В библиотеке PyTorch работа с данными организована через два ключевых компонента: `Dataset` и `DataLoader`.

Класс `Dataset` отвечает за представление набора данных и предоставляет интерфейс доступа к отдельным элементам выборки. Он реализует два основных метода:

- `__len__()` — возвращает размер набора данных;

- `__getitem__(index)` — возвращает объект данных и соответствующую метку по заданному индексу.

Такой подход позволяет описывать как стандартные наборы данных, так и пользовательские источники данных, например файлы изображений, текстовые документы или данные сенсоров.

Класс `DataLoader` используется для организации процесса подачи данных в модель. Он выполняет несколько важных функций:

- формирование батчей;
- случайное перемешивание данных;
- параллельную загрузку данных;
- итерацию по набору данных в процессе обучения.

Использование `DataLoader` значительно упрощает реализацию цикла обучения нейронной сети и делает код более структурированным и масштабируемым.

Правильная организация работы с данными является важным этапом разработки систем машинного обучения. Она позволяет повысить эффективность обучения, упростить масштабирование моделей и обеспечить корректную обработку больших наборов данных.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- NumPy
- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы студент должен реализовать подготовку данных и организовать процесс их передачи в модель с использованием механизмов `Dataset` и `DataLoader`.

На первом этапе необходимо подготовить набор данных. Это может быть как стандартный набор данных (например MNIST), так и искусственно сгенерированные данные для задачи классификации.

Далее следует создать объект класса `Dataset`, описывающий структуру набора данных. В случае пользовательского набора данных необходимо реализовать методы `__len__()` и `__getitem__()`.

После этого необходимо создать объект `DataLoader`, который будет использоваться для формирования батчей и передачи данных в модель. При создании загрузчика следует задать размер батча и режим перемешивания данных.

Следующим этапом является реализация цикла обучения модели, в котором данные извлекаются из `DataLoader` и передаются в нейронную сеть.

В процессе выполнения работы рекомендуется вывести информацию о структуре батчей, проверить размерности входных данных и убедиться в корректности работы механизма загрузки данных.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Подготовить простой набор данных для задачи классификации или регрессии.
2. Реализовать собственный класс набора данных, наследующий `Dataset`.
3. Реализовать методы `__len__()` и `__getitem__()` для доступа к элементам набора данных.
4. Создать объект `DataLoader` для формирования батчей и перемешивания данных.
5. Использовать `DataLoader` в цикле обучения модели нейронной сети.

Контрольные вопросы

1. Почему подготовка данных является важным этапом обучения нейронных сетей.
2. Для чего используется класс `Dataset` в библиотеке PyTorch.
3. Какие методы необходимо реализовать при создании пользовательского набора данных.
4. Какие функции выполняет класс `DataLoader`.
5. Что представляет собой батч при обучении нейронной сети.
6. Для чего используется перемешивание данных при обучении.
7. Чем отличается обучающая выборка от тестовой.
8. Как можно организовать работу с пользовательскими наборами данных.
9. Какие преимущества предоставляет использование `DataLoader` при обучении моделей.
10. Как проверить корректность структуры данных, передаваемых в модель.

ПРАКТИЧЕСКАЯ РАБОТА №5

Практическая работа №5

Тема: Реализация методов регуляризации нейронных сетей (Dropout, Batch Normalization)

Цель работы

Овладение навыками применения методов регуляризации нейронных сетей для повышения качества обучения и предотвращения переобучения модели. Изучение принципов работы методов Dropout и Batch Normalization и их реализации с использованием библиотеки PyTorch.

Теоретическое обоснование

Одной из основных проблем обучения нейронных сетей является переобучение. Переобучение возникает в ситуации, когда модель хорошо запоминает обучающую выборку, но плохо обобщает знания на новых данных. Это приводит к снижению точности модели при работе с тестовыми или реальными данными.

Для уменьшения эффекта переобучения применяются методы регуляризации. Регуляризация представляет собой совокупность методов, направленных на повышение способности модели обобщать знания и предотвращение чрезмерной адаптации к обучающим данным.

Одним из наиболее распространённых методов регуляризации является Dropout. Его идея заключается в случайном отключении части нейронов в процессе обучения. На каждой итерации обучения определённая доля нейронов временно исключается из вычислений. Это заставляет сеть формировать более устойчивые представления данных и снижает зависимость от отдельных нейронов.

Если обозначить вероятность отключения нейрона как p, то в процессе обучения выход нейрона может быть представлен следующим образом:

$$y = m \cdot a$$

где
a — выход нейрона,
m — случайная маска, принимающая значения 0 или 1.

Во время тестирования Dropout не применяется, а веса сети автоматически масштабируются для компенсации эффекта случайного отключения нейронов.

Другим важным методом является Batch Normalization. Он предназначен для нормализации распределения входных данных на каждом слое нейронной сети.

В процессе обучения вычисляются среднее значение и стандартное отклонение элементов батча:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

После этого выполняется нормализация:

$$\widehat{x_i} = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

где ε — малое число, предотвращающее деление на ноль.

Нормализованные значения затем масштабируются и сдвигаются с использованием обучаемых параметров.

Использование Batch Normalization ускоряет процесс обучения, стабилизирует работу нейронной сети и позволяет применять более высокие значения скорости обучения.

Библиотека PyTorch предоставляет готовые реализации данных методов через классы `torch.nn.Dropout` и `torch.nn.BatchNorm1d` (или `BatchNorm2d` для изображений).

Применение методов регуляризации является важным этапом разработки нейронных сетей, особенно при работе с ограниченными наборами данных.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- torchvision
- NumPy
- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы необходимо реализовать модель нейронной сети с использованием методов регуляризации.

На первом этапе следует реализовать базовую модель многослойного персептрона без использования методов регуляризации и обучить её на выбранном наборе данных (например MNIST).

После получения результатов необходимо модифицировать архитектуру модели, добавив слой Dropout между скрытыми слоями сети. Затем следует повторно обучить модель и сравнить результаты с исходной моделью.

Следующим этапом является добавление Batch Normalization в архитектуру сети. Нормализация должна выполняться после линейного слоя и перед функцией активации.

После реализации обоих методов регуляризации следует провести обучение модели и сравнить результаты работы сети в различных конфигурациях.

Особое внимание необходимо уделить анализу изменения точности модели и поведения функции потерь в процессе обучения.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Реализовать базовую модель многослойного персептрона без использования регуляризации.
2. Добавить слой Dropout в архитектуру модели и провести повторное обучение.
3. Реализовать использование Batch Normalization между слоями сети.
4. Сравнить точность моделей с регуляризацией и без неё.
5. Проанализировать влияние методов регуляризации на устойчивость обучения модели.

Контрольные вопросы

1. Что представляет собой переобучение нейронной сети.
2. Для чего используются методы регуляризации.
3. Как работает метод Dropout.
4. В какой момент обучения применяется Dropout.
5. Что представляет собой Batch Normalization.
6. Какие преимущества даёт нормализация данных в нейронных сетях.
7. В каком месте архитектуры сети применяется Batch Normalization.
8. Какие классы PyTorch используются для реализации Dropout и Batch Normalization.
9. Как регуляризация влияет на качество обобщения модели.
10. В каких случаях использование регуляризации особенно важно.

ПРАКТИЧЕСКАЯ РАБОТА №6

Практическая работа №6

Тема: Построение и обучение сверточной нейронной сети для классификации изображений

Цель работы

Овладение навыками построения и обучения сверточной нейронной сети для решения задачи классификации изображений. Изучение принципов работы сверточных слоев, операций подвыборки и методов построения архитектуры сверточных нейронных сетей с использованием библиотеки PyTorch.

Теоретическое обоснование

При решении задач компьютерного зрения традиционные полносвязные нейронные сети сталкиваются с рядом трудностей. Изображения содержат большое количество пикселей, а связи между соседними пикселями имеют пространственную структуру. Использование полностью связанных слоев приводит к резкому увеличению числа параметров и ухудшению эффективности обучения.

Сверточные нейронные сети (Convolutional Neural Networks, CNN) специально разработаны для обработки изображений и других данных с пространственной структурой. Основная идея CNN заключается в применении операций свертки, которые позволяют извлекать локальные признаки из изображения.

Операция свертки выполняется с использованием небольшого фильтра (ядра), который последовательно перемещается по изображению. В каждой позиции вычисляется скалярное произведение между элементами фильтра и соответствующей областью изображения. Результатом является карта признаков, отражающая наличие определённых структур в исходном изображении.

Математически операция свертки может быть представлена следующим образом:

$$y(i,j) = \sum_m \sum_n x(i+m, j+n) \cdot w(m,n)$$

где
x — входное изображение,
w — ядро свертки,
y — карта признаков.

После применения сверточного слоя обычно используется функция активации (например ReLU), которая вводит нелинейность в модель.

Для уменьшения размерности карт признаков и повышения устойчивости сети применяется операция подвыборки (Pooling). Наиболее распространённым видом является операция максимальной подвыборки (Max Pooling), при которой из каждой области выбирается максимальное значение.

Типичная архитектура сверточной нейронной сети включает следующие компоненты:

- сверточные слои для извлечения признаков;
- функции активации;

- слои подвыборки;
- полносвязные слои для окончательной классификации.

Сверточные нейронные сети широко используются в задачах распознавания изображений, анализа видео, медицинской диагностики, системах автономного управления и многих других областях.

Библиотека PyTorch предоставляет готовые инструменты для реализации сверточных нейронных сетей, включая классы `torch.nn.Conv2d`, `torch.nn.MaxPool2d` и другие элементы архитектуры.

В рамках данной работы студенты построят простую сверточную нейронную сеть и обучат её для классификации изображений.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- torchvision
- NumPy
- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы необходимо реализовать сверточную нейронную сеть и обучить её для решения задачи классификации изображений.

На первом этапе следует подготовить набор данных. Для обучения рекомендуется использовать набор изображений MNIST или другой аналогичный набор данных.

Далее необходимо реализовать архитектуру сверточной нейронной сети. В структуру модели должны входить сверточные слои (`Conv2d`), функции активации ReLU и слои подвыборки (`MaxPool2d`). После нескольких сверточных слоев необходимо добавить полносвязные слои для выполнения классификации.

Следующим этапом является определение функции потерь и оптимизатора. После этого необходимо реализовать цикл обучения, включающий прямое распространение данных через сеть, вычисление функции потерь, обратное распространение ошибки и обновление параметров модели.

После завершения обучения следует оценить точность модели на тестовой выборке и проанализировать результаты классификации.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Подготовить набор изображений для задачи классификации.
2. Реализовать архитектуру сверточной нейронной сети с использованием слоев `Conv2d` и `MaxPool2d`.
3. Реализовать полносвязные слои для классификации признаков, извлечённых сверточной частью сети.
4. Обучить модель на обучающей выборке и проанализировать изменение функции потерь.
5. Оценить точность модели на тестовой выборке и проанализировать результаты классификации.

Контрольные вопросы

1. Чем сверточные нейронные сети отличаются от полносвязных нейронных сетей.
2. Что представляет собой операция свертки.
3. Какую роль выполняют фильтры в сверточной сети.
4. Для чего используется операция подвыборки (Pooling).
5. Какие преимущества имеют сверточные нейронные сети при работе с изображениями.
6. Какие основные компоненты входят в архитектуру сверточной нейронной сети.
7. Какие классы PyTorch используются для реализации сверточных слоев.
8. Для чего используются полносвязные слои в конце сверточной сети.
9. Какие этапы включает процесс обучения сверточной нейронной сети.
10. В каких областях применяются сверточные нейронные сети.

ПРАКТИЧЕСКАЯ РАБОТА №7

Тема: Реализация рекуррентной нейронной сети для обработки последовательных данных

Цель работы

Овладение навыками построения и обучения рекуррентной нейронной сети для обработки последовательных данных. Изучение принципов работы рекуррентных архитектур и их реализации с использованием библиотеки PyTorch для решения задач анализа временных рядов и текстовых последовательностей.

Теоретическое обоснование

Во многих задачах машинного обучения данные имеют последовательную структуру. К таким данным относятся текстовые документы, временные ряды, сигналы, последовательности событий и другие виды информации, в которых важен порядок элементов.

Традиционные полносвязные нейронные сети не учитывают зависимость между элементами последовательности. Для обработки таких данных используются рекуррентные нейронные сети (Recurrent Neural Networks, RNN), которые обладают внутренним состоянием и способны учитывать информацию о предыдущих элементах последовательности.

Основная идея рекуррентной нейронной сети заключается в использовании скрытого состояния, которое передаётся от одного шага последовательности к другому. Это позволяет сети учитывать контекст предыдущих входных данных.

На каждом шаге последовательности вычисляется новое скрытое состояние:

$$h_t = f(W_x x_t + W_h h_{t-1} + b)$$

где
x_t — входной элемент последовательности в момент времени (t),
h_{t-1} — скрытое состояние предыдущего шага,
h_t — новое скрытое состояние,
W_x и W_h — матрицы весов,
b — вектор смещения,
$f(\cdot)$ — функция активации.

Выход сети может вычисляться на каждом шаге последовательности или только после обработки всей последовательности.

Рекуррентные нейронные сети широко применяются в задачах обработки естественного языка, анализа временных рядов, распознавания речи, машинного перевода и других задачах, где важен порядок элементов данных.

В практических системах используются различные модификации рекуррентных сетей, например LSTM (Long Short-Term Memory) и GRU (Gated Recurrent Unit), которые позволяют эффективнее работать с длинными последовательностями.

Библиотека PyTorch предоставляет готовые реализации рекуррентных слоёв через классы `torch.nn.RNN`, `torch.nn.LSTM` и `torch.nn.GRU`.

В рамках данной работы будет реализована простая рекуррентная нейронная сеть для обработки последовательных данных и обучения модели на примере задачи прогнозирования или классификации последовательностей.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- NumPy
- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы необходимо реализовать рекуррентную нейронную сеть и обучить её для обработки последовательных данных.

На первом этапе необходимо подготовить набор последовательных данных. Это может быть набор текстовых последовательностей, последовательность числовых значений временного ряда или синтетически сгенерированные последовательности.

Далее следует реализовать архитектуру рекуррентной нейронной сети с использованием одного из рекуррентных слоёв PyTorch, например `torch.nn.RNN`.

После определения архитектуры модели необходимо реализовать выходной слой, который будет формировать итоговое предсказание модели.

Следующим этапом является определение функции потерь и оптимизатора. Затем реализуется цикл обучения, включающий обработку последовательностей, вычисление функции потерь, обратное распространение ошибки и обновление параметров модели.

После завершения обучения необходимо оценить качество модели на тестовой выборке и проанализировать её способность учитывать зависимости между элементами последовательности.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Подготовить простой набор последовательных данных (например временной ряд или последовательность чисел).
2. Реализовать архитектуру рекуррентной нейронной сети с использованием слоя `torch.nn.RNN`.
3. Добавить выходной слой для получения итогового предсказания модели.
4. Реализовать цикл обучения модели на последовательных данных.

5. Проанализировать способность модели учитывать зависимости между элементами последовательности.

Контрольные вопросы

1. Какие задачи требуют обработки последовательных данных.
2. Чем рекуррентные нейронные сети отличаются от полносвязных сетей.
3. Что представляет собой скрытое состояние рекуррентной сети.
4. Как происходит обновление скрытого состояния в рекуррентной сети.
5. Какие проблемы возникают при обучении рекуррентных нейронных сетей.
6. Какие архитектуры используются для улучшения работы рекуррентных сетей.
7. Какие классы PyTorch используются для реализации рекуррентных слоёв.
8. В каких задачах применяются рекуррентные нейронные сети.
9. Какие этапы включает процесс обучения рекуррентной нейронной сети.
10. Как можно оценить качество модели при работе с последовательными данными.

ПРАКТИЕСКАЯ РАБОТА №8

Практическая работа №8

Тема: Использование механизма внимания и архитектуры Transformer в PyTorch

Цель работы

Овладение навыками использования механизма внимания (attention) и архитектуры Transformer для обработки последовательных данных. Изучение принципов работы механизмов self-attention и построения моделей Transformer с использованием средств библиотеки PyTorch.

Теоретическое обоснование

При обработке последовательных данных традиционные рекуррентные нейронные сети сталкиваются с рядом ограничений. Одной из основных проблем является сложность обучения на длинных последовательностях, поскольку информация постепенно теряется при передаче через множество временных шагов. Кроме того, рекуррентные сети плохо масштабируются при обработке больших наборов данных.

Для решения этих проблем был предложен механизм внимания (attention). Основная идея механизма внимания заключается в том, что модель может динамически определять, какие элементы входной последовательности являются наиболее важными при формировании текущего выхода.

Вместо последовательной обработки элементов данных механизм внимания позволяет вычислять взаимосвязи между всеми элементами последовательности одновременно. Это значительно повышает эффективность обучения и качество моделей.

Одним из наиболее распространённых видов внимания является механизм self-attention. Он позволяет каждому элементу последовательности учитывать информацию обо всех остальных элементах.

Математически механизм self-attention основан на вычислении трёх матриц: запросов (Query), ключей (Key) и значений (Value):

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$

где
X — входная последовательность,
W_Q, W_K, W_V — обучаемые матрицы весов.

Далее вычисляется матрица внимания:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

где d_k — размерность вектора ключей.

Архитектура Transformer была предложена в работе "Attention is All You Need" и основана полностью на механизме внимания без использования рекуррентных или сверточных слоёв.

Основные компоненты архитектуры Transformer включают:

- механизм multi-head attention;
- позиционное кодирование;
- полносвязные слои;
- нормализацию слоёв (Layer Normalization);
- остаточные соединения.

Transformer стал основой большинства современных систем обработки естественного языка, включая модели машинного перевода, генерации текста и большие языковые модели.

Библиотека PyTorch предоставляет готовые инструменты для работы с архитектурой Transformer, включая классы `torch.nn.MultiheadAttention`, `torch.nn.Transformer` и связанные компоненты.

В рамках данной работы студенты познакомятся с реализацией механизма внимания и реализуют простую модель Transformer для обработки последовательных данных.

Аппаратура и материалы

Персональный компьютер с установленной операционной системой Windows, macOS или Linux

Интерпретатор Python версии 3.8 и выше

Среда разработки: Jupyter Notebook, Google Colab или аналогичная

Установленные библиотеки:

- PyTorch
- NumPy

- Matplotlib

Методические указания по выполнению работы

В ходе выполнения работы необходимо реализовать модель, использующую механизм внимания и элементы архитектуры Transformer.

На первом этапе следует подготовить набор последовательных данных. Это может быть простой текстовый набор данных или синтетические последовательности числовых значений.

Далее необходимо реализовать механизм self-attention с использованием встроенных инструментов PyTorch, например класса `torch.nn.MultiheadAttention`.

После этого следует построить упрощённую архитектуру Transformer, включающую блок внимания и полносвязные слои.

Следующим этапом является определение функции потерь и оптимизатора. Затем необходимо реализовать цикл обучения модели, включающий прямое распространение данных через сеть, вычисление функции потерь, обратное распространение ошибки и обновление параметров модели.

После завершения обучения необходимо проанализировать способность модели учитывать зависимости между элементами последовательности и оценить качество предсказаний.

Требования по технике безопасности

Работа за компьютером должна выполняться в хорошо освещённом помещении.

Не рекомендуется непрерывная работа за компьютером более двух академических часов без перерыва.

Рабочее место должно быть организовано в соответствии с санитарными нормами.

Не допускается установка или изменение программного обеспечения без согласования с преподавателем.

Следует использовать только учебные наборы данных и избегать работы с конфиденциальной информацией.

Типовые задачи

1. Подготовить простой набор последовательных данных для обработки моделью.
2. Реализовать механизм self-attention с использованием класса `torch.nn.MultiheadAttention`.
3. Построить упрощённый блок архитектуры Transformer.
4. Реализовать обучение модели на последовательных данных.
5. Проанализировать результаты работы модели и оценить её способность учитывать зависимости между элементами последовательности.

Контрольные вопросы

1. Какие ограничения имеют рекуррентные нейронные сети при работе с длинными последовательностями.
2. В чём заключается идея механизма внимания.
3. Что представляют собой матрицы Query, Key и Value.
4. Как вычисляется механизм self-attention.
5. Что представляет собой архитектура Transformer.
6. Какие основные компоненты входят в архитектуру Transformer.
7. Для чего используется multi-head attention.
8. Какую роль играет позиционное кодирование в Transformer.
9. Какие классы PyTorch используются для реализации механизмов внимания.
10. В каких областях применяются модели на основе Transformer.

ЗАКЛЮЧЕНИЕ

Практические работы, представленные в рамках дисциплины «Основы нейронных сетей», являются важной частью образовательного процесса и направлены на формирование у студентов практических навыков разработки, обучения и анализа моделей искусственных нейронных сетей. Их выполнение обеспечивает переход от теоретического понимания принципов функционирования нейронных сетей к их практической реализации и применению при решении задач анализа данных.

В ходе выполнения практических заданий студенты знакомятся с ключевыми архитектурами нейронных сетей и этапами их построения: от реализации искусственного нейрона и однослойного персептрона до разработки многослойных моделей, сверточных и рекуррентных нейронных сетей, а также современных архитектур на основе механизма внимания. Особое внимание уделяется организации процесса обучения моделей, подготовке данных, использованию методов регуляризации и применению современных инструментов разработки на языке Python с использованием библиотеки PyTorch.

Комплекс практических работ способствует развитию алгоритмического мышления, навыков программной реализации моделей машинного обучения, способности анализировать поведение нейронных сетей и интерпретировать результаты их работы. Последовательное усложнение заданий позволяет студентам сформировать системное понимание архитектур нейронных сетей и принципов их применения в задачах классификации, обработки изображений и анализа последовательных данных.

Выполнение практических работ обеспечивает практико-ориентированное освоение методов построения нейронных сетей, способствует развитию компетенции ПК-1 и формирует готовность студентов к использованию современных технологий машинного обучения и нейросетевых моделей в профессиональной деятельности.

УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература:

Болотский, А. В.
    Исследование операций и методы оптимизации Электронный ресурс / Болотский А. В., Кочеткова О. А. - Санкт-Петербург : Лань, 2020. - 116 с. - ISBN 978-5-8114-4568-4

Сараев, П. В.
    Методы машинного обучения Электронный ресурс : Методические указания и задания к лабораторным работам по курсу / П. В. Сараев. - Липецк : Липецкий государственный технический университет, ЭБС АСВ, 2017. - 48 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 2227-8397

Сопов, Е. А.
    Многокритериальные нейроэволюционные системы в задачах машинного обучения и человеко-машинного взаимодействия Электронный

ресурс / Сопов Е. А., Иванов И. А. : монография. - Красноярск : СФУ, 2019. - 160 с. - ISBN 978-5-7638-3969-2

Дополнительная литература:

Методические рекомендации к практическим занятиям по дисциплине "Теория и практика машинного перевода" : Направление подготовки 45.04.02 Лингвистика. Магистерская программа "Перевод и переводоведение".Квалификация выпускника - магистр. Форма обучения - очная. Учебный план 2015 года. Изучается в 3 семестре. - Ставрополь : СКФУ, 2015. - 9 с.

Неделько, В. М.
    Основы статистических методов машинного обучения Электронный ресурс : Учебное пособие / В. М. Неделько. - Новосибирск : Новосибирский государственный технический университет, 2010. - 72 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 978-5-7782-1385-2

Рапаков, Г. Г.
    Методы и алгоритмы машинного обучения при принятии управленческих решений в региональной системе медицинской профилактики (опыт Вологодской области) Электронный ресурс / Рапаков Г. Г., Касимов Р. А. : монография. - Вологда : ВоГУ, 2014. - 143 с. - ISBN 978-5-87851-548-1

Учебно-методическая литература:

Сараев, П. В.
    Методы машинного обучения Электронный ресурс : Методические указания и задания к лабораторным работам по курсу / П. В. Сараев. - Липецк : Липецкий государственный технический университет, ЭБС АСВ, 2017. - 48 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 2227-8397

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по организации и проведению самостоятельной работы
по дисциплине «Основы нейронных сетей»
для студентов направления подготовки
11.03.02 Инфокоммуникационные технологии и системы связи
Направленность (профиль)
«Интеллектуальная обработка данных в инфокоммуникационных системах и сетях»

Ставрополь 2026

Содержание

ВВЕДЕНИЕ

1. Цель и задачи освоения дисциплины (модуля)

Цель дисциплины – формирование, согласно образовательной программы, компетенции ПК-1 обучающегося.

Цель дисциплины – формирование, согласно образовательной программы, компетенции ПК-1 обучающегося.

Для достижения цели освоения дисциплины «Основы нейронных сетей» в процессе обучения решаются следующие задачи: ознакомление с биологическими предпосылками возникновения искусственных нейронных сетей и формированием математической модели искусственного нейрона; изучение принципов построения однослойных и многослойных нейронных сетей, включая модель персептрона и её ограничения; освоение математических основ обучения нейронных сетей как задачи оптимизации, включая функции потерь для задач регрессии и классификации; изучение методов градиентной оптимизации, включая стохастический и мини-батч градиентный спуск; овладение современными методами ускорения обучения нейронных сетей, включая алгоритмы Momentum, AdaGrad, RMSprop, Adadelta, Adam и их модификации; понимание факторов, влияющих на сходимость и скорость обучения нейронных сетей; освоение практических навыков реализации и обучения нейронных сетей на языке Python с использованием библиотеки PyTorch; развитие умений применять нейронные сети для решения задач классификации и регрессии, возникающих при анализе данных и сигналов в инфокоммуникационных системах; формирование навыков проведения вычислительных экспериментов, анализа результатов обучения моделей и интерпретации полученных решений.

2. Место дисциплины (модуля) в структуре образовательной программы

Дисциплина относится к факультативным дисциплинам.

3. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с планируемыми результатами освоения образовательной программы

Код, формулировка компетенции	Код, формулировка индикатора	Планируемые результаты обучения по дисциплине (модулю), характеризующие этапы формирования компетенций, индикаторов
ПК-1. Способен к развитию коммутационных подсистем и сетевых платформ, сетей передачи данных, транспортных сетей и сетей радиодоступа, спутниковых систем связи	ИД-2$_{ПК-1}$ анализирует статистические параметры трафика, вырабатывает решения по оперативному переконфигурированию сети, изменению параметров коммутационной подсистемы, сетевых платформ, по применению оборудования новых технологий; изменению и корректировке параметров коммутационной подсистемы, транспортных сетей и сетей передачи данных, по организации новых и	Проведение анализа статистических параметров трафика; выработка решений по оперативному переконфигурированию сети; изменение параметров коммутационной подсистемы и сетевых платформ; применение оборудования новых технологий; изменение и корректировка параметров коммутационной

	расширению имеющихся направлений связи.	подсистемы, транспортных сетей и сетей передачи данных; организация новых и расширение имеющихся направлений связи.
	ИД-3$_{ПК-1}$. анализирует статистику основных показателей эффективности радиосистем и систем передачи данных, разрабатывает мероприятия по их поддержанию на требуемом уровне, выполняет расчет пропускной способности сетей телекоммуникаций.	Проведение анализа статистики основных показателей эффективности радиосистем и систем передачи данных; разработка мероприятий по поддержанию показателей эффективности радиосистем и систем передачи данных на требуемом уровне; выполнение расчета пропускной способности сетей телекоммуникаций.
	ИД-4$_{ПК-1}$ разрабатывает схемы организации связи и интеграции новых сетевых элементов, осуществляет построение и расширение коммутационной подсистемы и сетевых платформ; выполняет работы на коммутационном оборудовании, развертывает оборудование сервисных платформ, оборудование новых технологий на сети, выполняет планы по расширению существующего оборудования сетевых платформ и новых технологий.	Разработка схем организации связи и интеграции новых сетевых элементов; осуществление построения и расширения коммутационной подсистемы и сетевых платформ; выполнение работ на коммутационном оборудовании; развертывание оборудования сервисных платформ и оборудования новых технологий на сети; выполнение планов по расширению существующего оборудования сетевых платформ и новых технологий.

4. Объем учебной дисциплины (модуля) и формы контроля *

Объем занятий: всего: 3 з.е. 108 ак.ч.	ОФО, В акад. часах
Контактная работа:	24.00/18.00
Лекции/из них практическая подготовка	8.00/0
Лабораторных работ/из них практическая подготовка	-
Практических занятий/из них практическая подготовка	16.00/0
Самостоятельная работа	84.00

Формы контроля	
Экзамен	нет
Зачет	нет
Зачет с оценкой	да
Курсовая работа	нет

САМОСТОЯТЕЛЬНАЯ РАБОТА КАК ВАЖНЕЙШАЯ ФОРМА УЧЕБНОГО ПРОЦЕССА

Самостоятельная работа студентов является неотъемлемой частью освоения дисциплины «Основы нейронных сетей» и способствует формированию ключевых профессиональных компетенций. В условиях стремительного развития технологий ИИ и их внедрения в различные отрасли экономики и социальной сферы, способность к самостоятельному освоению новых инструментов, подходов и методов приобретает первостепенное значение.

Изучение дисциплины требует от студентов активного включения в образовательный процесс: поиска и анализа информации, выполнения практико-ориентированных заданий, работы с программными средствами, анализа прикладных кейсов. Самостоятельная работа способствует развитию аналитического и системного мышления, умения принимать решения в условиях неопределённости, формулировать требования к интеллектуальным системам, а также применять технологии ИИ в конкретной профессиональной среде.

Особое значение имеет развитие навыков проектирования интеллектуальных решений с учётом специфики предметной области, что требует от студентов глубокого понимания контекста, изучения отраслевых примеров и способности к обоснованному выбору алгоритмов и моделей.

Таким образом, самостоятельная работа студентов формирует устойчивую базу для профессионального саморазвития, гибкости мышления и готовности к решению задач, выходящих за рамки типовых сценариев, что особенно важно для специалистов, взаимодействующих с технологиями искусственного интеллекта на практике.

Самостоятельная работа - планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное (аудиторное) время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия (при частичном непосредственном участии преподавателя, оставляющем ведущую роль за работой студентов).

Самостоятельная работа студентов в ВУЗе является важным видом учебной и научной деятельности студента. Самостоятельная работа студентов играет значительную роль в рейтинговой технологии обучения. Обучение в ВУЗе включает в себя две, практически одинаковые по объему и взаимовлиянию части – процесса обучения и процесса самообучения. Поэтому СРС должна стать эффективной и целенаправленной работой студента.

Концепцией модернизации российского образования определены основные задачи профессионального образования - "подготовка квалифицированного работника соответствующего уровня и профиля, конкурентоспособного на рынке труда, компетентного, ответственного, свободно владеющего своей профессией и ориентированного в смежных областях деятельности, способного к эффективной работе по специальности на уровне мировых стандартов, готового к постоянному профессиональному росту, социальной и профессиональной мобильности".

Решение этих задач невозможно без повышения роли самостоятельной работы студентов над учебным материалом, усиления ответственности преподавателей за развитие навыков самостоятельной работы, за стимулирование профессионального роста студентов, воспитание творческой активности и инициативы.

К современному специалисту общество предъявляет достаточно широкий перечень требований, среди которых немаловажное значение имеет наличие у выпускников определенных способностей и умения самостоятельно добывать знания из различных источников, систематизировать полученную информацию, давать оценку конкретной финансовой ситуации. Формирование такого умения происходит в течение всего периода обучения через участие студентов в лабораторных занятиях, выполнение контрольных заданий и тестов, написание курсовых и выпускных квалификационных работ. При этом самостоятельная работа студентов играет решающую роль в ходе всего учебного процесса.

Формы самостоятельной работы студентов разнообразны. По данной дисциплине «Искусственный интеллект и машинное обучение» предусмотрены следующие:

Использование материала учебно-методического комплекса дисциплины

На первом этапе необходимо ознакомиться с обязательным минимумом содержания основной образовательной программы по изучаемой дисциплине. Далее необходимо изучить рабочую программу дисциплины, в которой рассмотрено содержание тем дисциплины лекционного курса, взаимосвязь тем лекций с лабораторных занятий, темы и виды самостоятельной работы. По каждому виду самостоятельной работы предусмотрены определённые формы отчетности, представленные в учебной программе по дисциплине «Искусственный интеллект и машинное обучение».

Работа с литературой

Для успешного освоения дисциплины, необходимо самостоятельно детально изучить представленные темы по рекомендуемым источникам информации, которые представлены в учебной программе.

Самостоятельная работа приобщает студентов к научному творчеству, поиску и решению актуальных современных проблем.

ЦЕЛИ И ОСНОВНЫЕ ЗАДАЧИ СРС

Цель самостоятельной работы

Формирование у студентов устойчивых знаний и навыков, направленных на применение методов и технологий искусственного интеллекта в профессиональной деятельности. Развитие способности к самостоятельному поиску, отбору и адаптации ИИ-решений с учётом специфики предметной области.

Задачи самостоятельной работы

- Расширение и углубление теоретических знаний, полученных на лекциях.
- Формирование навыков работы с информационными и программными ресурсами по тематике дисциплины.
- Овладение методами анализа профессиональных задач с позиций применимости искусственного интеллекта.
- Развитие умений проектировать и аргументированно обосновывать ИИ-решения.
- Формирование компетенций в области выбора и применения инструментальных средств (библиотек, моделей, API).

Формы организации самостоятельной работы

- Выполнение индивидуальных заданий по ключевым темам дисциплины.

- Анализ профессиональных кейсов и проектных ситуаций.
- Работа с документацией к библиотекам
- Подготовка аналитических записок, презентаций и мини-отчетов.
- Выполнение тестовых заданий, в том числе с элементами самоконтроля.

Ведущая цель организации и осуществления СРС должна совпадать с целью обучения студента – подготовкой бакалавра с высшим образованием. При организации СРС важным и необходимым условием становятся формирование умения самостоятельной работы для приобретения знаний, навыков и возможности организации учебной и научной деятельности.

Целью самостоятельной работы студентов является овладение фундаментальными знаниями, профессиональными умениями и навыками деятельности по профилю, опытом творческой, исследовательской деятельности. Самостоятельная работа студентов способствует развитию самостоятельности, ответственности и организованности, творческого подхода к решению проблем учебного и профессионального уровня.

Задачами СРС являются:

- систематизация и закрепление полученных теоретических знаний и практических умений студентов;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;
- развитие исследовательских умений;
- использование материала, собранного и полученного в ходе самостоятельных занятий на семинарах, на лабораторных занятиях, при написании курсовых и выпускной квалификационной работ, для эффективной подготовки к итоговым зачетам и экзаменам.

ВИДЫ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

В образовательном процессе высшего профессионального образовательного учреждения выделяется два вида самостоятельной работы – аудиторная, под руководством преподавателя, и внеаудиторная. Тесная взаимосвязь этих видов работ предусматривает дифференциацию и эффективность результатов ее выполнения и зависит от организации, содержания, логики учебного процесса (межпредметных связей, перспективных знаний и др.):

Аудиторная самостоятельная работа по дисциплине выполняется на учебных занятиях под непосредственным руководством преподавателя и по его заданию.

Внеаудиторная самостоятельная работа выполняется студентом по заданию преподавателя, но без его непосредственного участия.

Основными видами самостоятельной работы студентов без участия преподавателя по данной дисциплине являются:

- формирование и усвоение содержания конспекта лекций на базе рекомендованной лектором учебной литературы, включая информационные образовательные ресурсы (электронные учебники, электронные библиотеки и др.);
- подготовка к лабораторным работам, их оформление.

Основными видами самостоятельной работы студентов с участием преподавателей являются:

- текущие консультации;
- прием и защита лабораторных работ (во время проведения л/р).

ОРГАНИЗАЦИЯ СРС

Методика организации самостоятельной работы студентов зависит от структуры, характера и особенностей дисциплины «Основы нейронных сетей», объема часов на ее изучение, вида заданий для самостоятельной работы студентов, индивидуальных качеств студентов и условий учебной деятельности.

Процесс организации самостоятельной работы студентов включает в себя следующие этапы:

- подготовительный (определение целей, составление программы, подготовка методического обеспечения, подготовка оборудования);
- основной (реализация программы, использование приемов поиска информации, усвоения, переработки, применения, передачи знаний, фиксирование результатов, самоорганизация процесса работы);
- заключительный (оценка значимости и анализ результатов, их систематизация, оценка эффективности программы и приемов работы, выводы о направлениях оптимизации труда).

Рекомендуемые темы и задания для самостоятельной проработки:

1. Основы работы с тензорами
- Изучить понятие тензора, размерность и операции над тензорами.
- Задание: создать несколько тензоров в PyTorch, выполнить операции сложения, умножения, транспонирования и вычисления среднего значения.
2. Генерация синтетических данных
- Изучить методы генерации искусственных наборов данных для задач классификации.
- Задание: сгенерировать набор данных с помощью sklearn.datasets.make_classification, визуализировать его и подготовить для обучения модели.
3. Визуализация данных
- Изучить методы визуализации распределения данных.
- Задание: построить гистограммы, scatter-графики и диаграммы плотности для выбранного набора данных.
4. Реализация линейной регрессии
- Изучить принцип работы линейной регрессии.
- Задание: реализовать модель линейной регрессии с использованием PyTorch и обучить её на синтетических данных.
5. Реализация логистической регрессии
- Изучить задачу бинарной классификации.
- Задание: реализовать логистическую регрессию и обучить модель на наборе данных breast_cancer.
6. Реализация искусственного нейрона
- Изучить структуру искусственного нейрона.
- Задание: реализовать модель нейрона с функцией активации sigmoid и проверить её работу на простом наборе данных.
7. Функции активации
- Изучить различные функции активации.
- Задание: реализовать и сравнить функции ReLU, sigmoid и tanh на одном и том же наборе данных.
8. Реализация однослойного персептрона
- Изучить принцип работы персептрона.

- Задание: реализовать однослойный персептрон для задачи классификации точек на плоскости.

9. Визуализация границы классификации
- Изучить понятие разделяющей гиперплоскости.
- Задание: визуализировать границу решения обученной модели классификации.

10. Функции потерь
- Изучить различные функции потерь.
- Задание: сравнить значения MSELoss и CrossEntropyLoss на одинаковом наборе данных.

11. Оптимизаторы нейронных сетей
- Изучить различные алгоритмы оптимизации.
- Задание: обучить модель с использованием SGD и Adam и сравнить скорость обучения.

12. Градиентный спуск
- Изучить принцип градиентного спуска.
- Задание: реализовать простой градиентный спуск для минимизации функции.

13. Минибатч обучение
- Изучить понятие батчей.
- Задание: реализовать обучение модели с использованием разных размеров батча.

14. Подготовка обучающих и тестовых выборок
- Изучить методы разделения данных.
- Задание: разделить набор данных на обучающую и тестовую части и оценить точность модели.

15. Использование Dataset
- Изучить структуру класса Dataset.
- Задание: реализовать собственный класс Dataset для хранения числовых данных.

16. Использование DataLoader
- Изучить возможности DataLoader.
- Задание: реализовать загрузку данных батчами и проверить их структуру.

17. Нормализация данных
- Изучить методы нормализации признаков.
- Задание: сравнить обучение модели с нормализованными и ненормализованными данными.

18. Переобучение моделей
- Изучить признаки переобучения.
- Задание: обучить модель с большим числом параметров и проанализировать её поведение.

19. Регуляризация L2
- Изучить методы регуляризации.
- Задание: обучить модель с использованием L2-регуляризации.

20. Dropout
- Изучить принцип работы Dropout.
- Задание: добавить слой Dropout в архитектуру сети и сравнить результаты обучения.

21. Batch Normalization
- Изучить метод Batch Normalization.
- Задание: добавить BatchNorm в модель и сравнить стабильность обучения.

22. Визуализация процесса обучения
- Изучить методы анализа обучения модели.
- Задание: построить графики функции потерь и точности по эпохам.

23. Подбор гиперпараметров
- Изучить влияние гиперпараметров на обучение.
- Задание: изменить скорость обучения и сравнить результаты.

24. Анализ ошибок классификации
- Изучить методы анализа ошибок модели.
- Задание: определить примеры, на которых модель ошибается.

25. Работа с набором данных MNIST
- Изучить структуру набора MNIST.
- Задание: загрузить набор данных и визуализировать изображения цифр.

26. Реализация MLP для MNIST
- Изучить архитектуру многослойного персептрона.
- Задание: реализовать MLP и обучить его на MNIST.

27. Анализ результатов классификации
- Изучить метрики качества.
- Задание: вычислить accuracy и confusion matrix.

28. Визуализация весов сети
- Изучить распределение параметров модели.
- Задание: визуализировать веса первого слоя сети.

29. Сверточные слои
- Изучить операцию свертки.
- Задание: применить фильтр свертки к изображению.

30. Реализация сверточной нейронной сети
- Изучить архитектуру CNN.
- Задание: реализовать простую CNN для MNIST.

31. Слои подвыборки
- Изучить MaxPooling.
- Задание: применить MaxPool к карте признаков.

32. Сравнение CNN и MLP
- Изучить различия архитектур.
- Задание: сравнить точность CNN и MLP на MNIST.

33. Визуализация карт признаков
- Изучить интерпретацию CNN.
- Задание: вывести карты признаков после сверточного слоя.

34. Работа с цветными изображениями
- Изучить структуру RGB изображений.
- Задание: загрузить набор CIFAR-10 и визуализировать изображения.

35. Предобработка изображений
- Изучить преобразования изображений.
- Задание: применить нормализацию и изменение размера изображений.

36. Обработка последовательностей
- Изучить задачи анализа последовательных данных.
- Задание: подготовить набор числовых последовательностей.

37. Реализация простой RNN
- Изучить принцип рекуррентных сетей.
- Задание: реализовать модель torch.nn.RNN.

38. Прогнозирование временного ряда
- Изучить задачи прогнозирования.
- Задание: обучить RNN для предсказания следующего значения последовательности.

39. Использование LSTM
- Изучить архитектуру LSTM.
- Задание: реализовать модель torch.nn.LSTM.

40. Сравнение RNN и LSTM
- Изучить различия архитектур.
- Задание: сравнить качество прогнозирования.

41. Кодирование текстовых данных

- Изучить методы кодирования текста.
- Задание: преобразовать текст в последовательность индексов.

42. Обучение модели для анализа текста
- Изучить задачи классификации текста.
- Задание: обучить простую модель для определения категории текста.

43. Механизм внимания
- Изучить принцип attention.
- Задание: реализовать простой механизм внимания.

44. Self-attention
- Изучить механизм self-attention.
- Задание: вычислить матрицу внимания для последовательности.

45. Multi-head attention
- Изучить архитектуру multi-head attention.
- Задание: реализовать слой MultiheadAttention.

46. Архитектура Transformer
- Изучить структуру Transformer.
- Задание: реализовать простой блок Transformer.

47. Позиционное кодирование
- Изучить методы кодирования позиции.
- Задание: реализовать синусоидальное позиционное кодирование.

48. Сравнение моделей последовательностей
- Изучить эффективность различных архитектур.
- Задание: сравнить RNN, LSTM и Transformer.

49. Сохранение и загрузка моделей
- Изучить методы сохранения моделей.
- Задание: сохранить обученную модель и загрузить её для повторного использования.

50. Разработка собственного проекта
- Изучить полный цикл разработки модели.
- Задание: выбрать задачу классификации или прогнозирования, подготовить данные, обучить модель и провести анализ результатов.

Типовые задания для СРС:

Общий подход к оцениванию:

- Каждое задание оценивается по **10-балльной шкале**;
- Общая оценка по блоку самостоятельной работы может рассчитываться как среднее значение или сумма (по усмотрению преподавателя);
- Возможно использование **частичных баллов (0.5, 1 и т.д.)** при частично выполненных пунктах задания;
- При копировании без осмысления, неработающем коде или отсутствующей интерпретации баллы **не начисляются**.

№	Задание	Критерии оценки (макс. 10 баллов)
1	Изучить структуру тензоров в PyTorch и выполнить базовые операции над ними.	корректное создание тензоров – 3; выполнение операций (сложение, умножение, reshape) – 4; вывод результатов и комментарии – 3
2	Сгенерировать синтетический набор данных для задачи классификации и визуализировать его.	корректная генерация данных – 3; визуализация данных – 4; интерпретация результатов – 3
3	Построить графики распределения	корректная загрузка данных – 3; построение

№	Задание	Критерии оценки (макс. 10 баллов)
	признаков выбранного набора данных.	графиков – 4; краткий анализ распределений – 3
4	Реализовать модель линейной регрессии в PyTorch.	корректная реализация модели – 4; обучение модели – 3; анализ результатов – 3
5	Реализовать логистическую регрессию для бинарной классификации.	реализация модели – 4; обучение и настройка параметров – 3; анализ точности – 3
6	Реализовать искусственный нейрон с функцией активации.	корректная формула нейрона – 4; реализация функции активации – 3; тестирование модели – 3
7	Сравнить функции активации ReLU, sigmoid и tanh.	корректная реализация функций – 3; проведение эксперимента – 4; анализ результатов – 3
8	Реализовать однослойный персептрон для классификации данных.	корректная архитектура – 4; обучение модели – 3; анализ результатов – 3
9	Визуализировать границу классификации обученной модели.	корректное обучение модели – 3; построение границы – 4; интерпретация – 3
10	Исследовать влияние различных функций потерь.	корректное применение функций потерь – 4; проведение эксперимента – 3; анализ результатов – 3
11	Сравнить оптимизаторы SGD и Adam.	реализация эксперимента – 4; корректное обучение – 3; анализ различий – 3
12	Реализовать алгоритм градиентного спуска для простой функции.	корректная реализация алгоритма – 4; демонстрация работы – 3; объяснение результатов – 3
13	Исследовать влияние размера батча на обучение модели.	корректная реализация эксперимента – 4; сравнение результатов – 3; выводы – 3
14	Разделить данные на обучающую и тестовую выборки.	корректное разделение данных – 4; обучение модели – 3; оценка качества – 3
15	Реализовать собственный класс Dataset.	корректная структура класса – 4; реализация методов – 3; тестирование – 3
16	Использовать DataLoader для загрузки данных.	корректное создание DataLoader – 4; работа с батчами – 3; анализ результатов – 3
17	Исследовать влияние нормализации данных на обучение.	корректная нормализация – 4; проведение эксперимента – 3; анализ – 3
18	Исследовать переобучение модели.	корректная реализация модели – 4; выявление переобучения – 3; анализ – 3
19	Реализовать L2-регуляризацию.	корректная реализация – 4; обучение модели – 3; анализ влияния – 3
20	Использовать Dropout для регуляризации сети.	корректное добавление слоя – 4; обучение модели – 3; анализ – 3
21	Использовать Batch Normalization в сети.	корректная реализация – 4; обучение – 3; анализ результатов – 3
22	Построить графики функции потерь и точности по эпохам.	корректное обучение – 3; построение графиков – 4; анализ – 3
23	Исследовать влияние скорости обучения.	корректный эксперимент – 4; сравнение результатов – 3; выводы – 3
24	Провести анализ ошибок классификации модели.	корректная оценка модели – 3; анализ ошибок – 4; выводы – 3

№	Задание	Критерии оценки (макс. 10 баллов)
25	Загрузить и визуализировать изображения MNIST.	корректная загрузка – 3; визуализация – 4; описание – 3
26	Реализовать MLP для классификации MNIST.	корректная архитектура – 4; обучение – 3; оценка точности – 3
27	Построить матрицу ошибок классификации.	корректное построение – 4; анализ результатов – 3; интерпретация – 3
28	Визуализировать веса первого слоя нейронной сети.	корректное извлечение весов – 4; визуализация – 3; анализ – 3
29	Применить свёрточный фильтр к изображению.	корректная реализация – 4; визуализация результата – 3; анализ – 3
30	Реализовать сверточную нейронную сеть для MNIST.	корректная архитектура – 4; обучение модели – 3; оценка точности – 3
31	Исследовать работу слоя MaxPooling.	корректная реализация – 4; демонстрация работы – 3; анализ – 3
32	Сравнить результаты MLP и CNN.	корректный эксперимент – 4; сравнение результатов – 3; выводы – 3
33	Визуализировать карты признаков сверточной сети.	корректное извлечение карт – 4; визуализация – 3; анализ – 3
34	Загрузить и визуализировать изображения CIFAR-10.	корректная загрузка – 3; визуализация – 4; описание – 3
35	Выполнить предобработку изображений (нормализация, изменение размера).	корректная реализация – 4; применение преобразований – 3; анализ – 3
36	Подготовить набор последовательных данных.	корректная подготовка – 4; описание структуры данных – 3; анализ – 3
37	Реализовать рекуррентную нейронную сеть (RNN).	корректная архитектура – 4; обучение модели – 3; анализ результатов – 3
38	Реализовать прогнозирование временного ряда с помощью RNN.	корректная реализация – 4; обучение – 3; оценка качества – 3
39	Реализовать модель LSTM.	корректная архитектура – 4; обучение – 3; анализ результатов – 3
40	Сравнить RNN и LSTM на задаче прогнозирования.	корректный эксперимент – 4; сравнение результатов – 3; выводы – 3
41	Реализовать кодирование текстовых данных.	корректная подготовка текста – 4; преобразование в последовательности – 3; описание – 3
42	Обучить модель для классификации текста.	корректная архитектура – 4; обучение модели – 3; оценка результатов – 3
43	Реализовать простой механизм внимания.	корректная реализация – 4; демонстрация работы – 3; анализ – 3
44	Реализовать self-attention для последовательности.	корректная формула внимания – 4; программная реализация – 3; анализ – 3
45	Использовать MultiheadAttention в PyTorch.	корректная реализация – 4; обучение модели – 3; анализ результатов – 3
46	Реализовать простой блок Transformer.	корректная архитектура – 4; обучение модели – 3; анализ – 3
47	Реализовать позиционное кодирование.	корректная реализация – 4; интеграция в модель – 3; анализ – 3

№	Задание	Критерии оценки (макс. 10 баллов)
48	Сравнить RNN, LSTM и Transformer для обработки последовательностей.	корректный эксперимент – 4; сравнение результатов – 3; выводы – 3
49	Реализовать сохранение и загрузку обученной модели.	корректное сохранение – 3; корректная загрузка – 4; проверка работы – 3
50	Выполнить мини-проект по применению нейронной сети для выбранной задачи.	корректная подготовка данных – 3; реализация и обучение модели – 4; анализ результатов – 3

Типовые задания для творческих групповых заданий СРС:

1. Архитектура ML-проекта
   Предложить структуру каталогов проекта машинного обучения (данные, модели, обучение, инференс, эксперименты). Обосновать назначение каждого каталога.
2. Документирование проекта
   Разработать файл README для проекта нейронной сети, описывающий цель проекта, структуру данных, порядок обучения модели и запуск инференса.
3. Конфигурация экспериментов
   Создать конфигурационный файл (например YAML или JSON) для хранения параметров обучения модели и описать, как его использовать в коде.
4. Управление версиями моделей
   Разработать систему версионирования моделей (например model_v1, model_v2). Описать правила хранения и обновления моделей.
5. Логирование процесса обучения
   Предложить механизм логирования обучения модели (например сохранение метрик в файл). Реализовать пример логирования.
6. Сравнение моделей
   Разработать систему сравнения нескольких обученных моделей на одном наборе данных.
7. Сохранение модели
   Реализовать сохранение обученной модели нейронной сети в файл и описать структуру сохраняемых данных.
8. Загрузка модели
   Написать код загрузки модели и проверки её работоспособности на тестовых данных.
9. Экспорт модели
   Исследовать возможности экспорта модели (например TorchScript или ONNX) и описать преимущества такого подхода.
10. Скрипт инференса
    Разработать отдельный скрипт для выполнения инференса модели на новых данных.
11. API для инференса
    Предложить архитектуру простого API (например REST-сервис) для выполнения инференса нейросетевой модели.
12. Оценка времени инференса
    Измерить время выполнения инференса для одного и нескольких объектов данных.
13. Оптимизация инференса
    Исследовать способы ускорения инференса (например уменьшение размера модели).
14. Анализ памяти модели
    Определить количество параметров модели и оценить объём занимаемой памяти.
15. Пакетная обработка данных
    Реализовать инференс модели для обработки данных пакетами.

16. Подготовка модели к внедрению
    Разработать план внедрения модели в информационную систему.
17. Создание Docker-контейнера
    Предложить структуру Docker-контейнера для развёртывания модели.
18. Разработка интерфейса пользователя
    Предложить идею простого пользовательского интерфейса для демонстрации работы модели.
19. Автоматизация обучения
    Разработать сценарий автоматического запуска обучения модели.
20. Мониторинг работы модели
    Предложить механизм мониторинга качества модели после внедрения.
21. Обработка ошибок инференса
    Разработать систему обработки ошибок при выполнении инференса.
22. Анализ стабильности модели
    Проверить устойчивость модели к небольшим изменениям входных данных.
23. Подготовка данных для инференса
    Разработать функцию предварительной обработки входных данных.
24. Постобработка результатов
    Реализовать процедуру обработки результатов работы модели перед выводом пользователю.
25. Сравнение производительности CPU и GPU
    Исследовать влияние аппаратного обеспечения на скорость инференса.
26. Оптимизация размера модели
    Исследовать методы уменьшения размера модели.
27. Тестирование модели
    Разработать набор тестов для проверки корректности работы модели.
28. Документация API модели
    Создать описание API для сервиса инференса.
29. Организация экспериментов
    Предложить систему хранения результатов экспериментов.
30. Воспроизводимость экспериментов
    Разработать механизм воспроизводимости обучения модели.
31. Анализ зависимостей проекта
    Составить список используемых библиотек и их назначение.
32. Подготовка модели для мобильных устройств
    Исследовать требования к моделям, используемым на мобильных устройствах.
33. Сжатие модели
    Изучить методы уменьшения числа параметров модели.
34. Создание демо-приложения
    Разработать концепцию демонстрационного приложения для модели.
35. Интерпретация результатов модели
    Предложить способы объяснения решений модели пользователю.
36. Анализ ошибок модели
    Провести анализ наиболее частых ошибок модели.
37. Создание отчёта по модели
    Подготовить технический отчёт по разработанной модели.
38. Подготовка модели для облачного сервиса
    Предложить архитектуру развёртывания модели в облаке.
39. Масштабирование инференса
    Разработать схему обработки большого количества запросов.
40. Разработка CLI-интерфейса
    Создать консольный интерфейс для запуска инференса модели.

41. Анализ требований к инфраструктуре
    Определить требования к аппаратному обеспечению для работы модели.
42. Интеграция модели с существующей системой
    Предложить способ интеграции модели в существующее программное обеспечение.
43. План тестирования модели
    Разработать план тестирования перед внедрением.
44. Проверка корректности данных
    Разработать систему проверки входных данных перед инференсом.
45. Разработка пайплайна обработки данных
    Предложить последовательность этапов обработки данных.
46. Автоматическое обновление модели
    Разработать концепцию обновления модели при появлении новых данных.
47. Анализ жизненного цикла модели
    Описать этапы жизненного цикла модели машинного обучения.
48. Разработка политики хранения моделей
    Определить правила хранения и удаления устаревших моделей.
49. Оценка рисков внедрения модели
    Проанализировать возможные риски внедрения нейросетевой системы.
50. Проектирование ML-сервиса
    Разработать архитектуру полноценного сервиса машинного обучения с этапами обучения, инференса и мониторинга.

Деятельность студентов по формированию и развитию навыков учебной самостоятельной работы

В процессе самостоятельной работы студент приобретает навыки самоорганизации, самоконтроля, самоуправления, саморефлексии и становится активным самостоятельным субъектом учебной деятельности.

Выполняя самостоятельную работу под контролем преподавателя студент должен:

– освоить минимум содержания, выносимый на самостоятельную работу студентов и предложенный преподавателем в соответствии с образовательными стандартами высшего образования;

– планировать самостоятельную работу в соответствии с графиком самостоятельной работы, предложенным преподавателем;

– самостоятельную работу студент должен осуществлять в организационных формах, предусмотренных учебным планом и рабочей программой преподавателя;

– выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

студент может:

сверх предложенного преподавателем (при обосновании и согласовании с ним) и минимума обязательного содержания, определяемого программой по данной дисциплине:

– самостоятельно определять уровень (глубину) проработки содержания материала;

– предлагать темы и вопросы для самостоятельной проработки;

– в рамках общего графика выполнения самостоятельной работы предлагать обоснованный индивидуальный график выполнения и отчетности по результатам самостоятельной работы;

– предлагать свои варианты организационных форм самостоятельной работы;

– использовать для самостоятельной работы методические пособия, учебные пособия, разработки сверх предложенного преподавателем перечня;

– использовать не только контроль, но и самоконтроль результатов самостоятельной работы в соответствии с методами самоконтроля, предложенными преподавателем или выбранными самостоятельно.

Самостоятельная работа студентов должна оказывать важное влияние на формирование личности будущего бакалавра, она планируется студентом самостоятельно. Каждый студент самостоятельно определяет режим своей работы и меру труда, затрачиваемого на овладение учебным содержанием по каждой дисциплине. Он выполняет внеаудиторную работу по личному индивидуальному плану, в зависимости от его подготовки, времени и других условий.

Вопросы для проверки СРС

1. Что понимается под искусственной нейронной сетью?
2. Какие задачи решаются с помощью нейронных сетей?
3. Чем нейронные сети отличаются от традиционных алгоритмов машинного обучения?
4. Какие основные этапы разработки нейросетевой модели?
5. Что представляет собой искусственный нейрон?
6. Какие элементы входят в структуру искусственного нейрона?
7. Что такое вес нейронной связи?
8. Какую роль играет смещение (bias) в нейроне?
9. Что такое функция активации?
10. Какие функции активации используются в нейронных сетях?
11. Чем отличаются функции ReLU, sigmoid и tanh?
12. Что представляет собой персептрон?
13. В чем отличие однослойного персептрона от многослойного?
14. Почему однослойный персептрон не может решать нелинейно разделимые задачи?
15. Что представляет собой многослойный персептрон?
16. Что такое скрытый слой нейронной сети?
17. Как определяется количество нейронов в слое?
18. Что такое архитектура нейронной сети?
19. Какие параметры нейронной сети обучаются?
20. Что представляет собой обучение нейронной сети?
21. Что такое обучающая выборка?
22. Что такое тестовая выборка?
23. Чем отличается обучающая выборка от валидационной?
24. Что такое признаки и метки в наборе данных?
25. Что такое нормализация данных?
26. Для чего используется масштабирование признаков?
27. Что такое one-hot кодирование?
28. Какие проблемы возникают при работе с несбалансированными данными?
29. Что такое переобучение модели?
30. Какие признаки указывают на переобучение?
31. Что такое недообучение модели?
32. Как можно уменьшить переобучение нейронной сети?
33. Что представляет собой регуляризация?
34. Что такое L1 и L2 регуляризация?
35. Что представляет собой метод Dropout?
36. Как работает Batch Normalization?
37. Что такое функция потерь?
38. Какие функции потерь применяются в задачах классификации?
39. Какие функции потерь применяются в задачах регрессии?

40. Что измеряет функция потерь?
41. Что такое градиент?
42. Что представляет собой алгоритм градиентного спуска?
43. Какие разновидности градиентного спуска существуют?
44. Что такое скорость обучения (learning rate)?
45. Как скорость обучения влияет на процесс обучения?
46. Что такое эпоха обучения?
47. Что такое батч при обучении нейронной сети?
48. Что такое минибатч обучение?
49. Что представляет собой обратное распространение ошибки?
50. Как вычисляются градиенты в нейронной сети?
51. Что такое оптимизатор?
52. Какие оптимизаторы используются при обучении нейронных сетей?
53. Чем отличается SGD от Adam?
54. Что такое моментум в оптимизации?
55. Как происходит обновление весов модели?
56. Что такое метрики качества модели?
57. Какие метрики используются для оценки классификации?
58. Что такое accuracy?
59. Что такое precision и recall?
60. Что такое матрица ошибок?
61. Что такое PyTorch?
62. Какие основные компоненты библиотеки PyTorch?
63. Что представляет собой тензор в PyTorch?
64. Чем тензор отличается от массива NumPy?
65. Какие операции можно выполнять над тензорами?
66. Что представляет собой модуль torch.nn?
67. Для чего используется класс torch.nn.Module?
68. Что представляет собой функция forward?
69. Как реализуется модель нейронной сети в PyTorch?
70. Как происходит обучение модели в PyTorch?
71. Что представляет собой Dataset в PyTorch?
72. Какие методы должен реализовывать класс Dataset?
73. Для чего используется DataLoader?
74. Какие параметры имеет DataLoader?
75. Что такое перемешивание данных (shuffle)?
76. Что такое карта признаков (feature map)?
77. Что представляет собой сверточная нейронная сеть?
78. Какие задачи решаются с помощью CNN?
79. Что такое операция свертки?
80. Что представляет собой фильтр свертки?
81. Что такое ядро свертки?
82. Как работает слой Conv2D?
83. Что представляет собой слой MaxPooling?
84. Для чего используется операция подвыборки?
85. Какие преимущества имеют сверточные сети?
86. Что такое рекуррентная нейронная сеть?
87. Какие задачи решаются с помощью RNN?
88. Что представляет собой скрытое состояние в RNN?
89. Какие проблемы возникают при обучении RNN?
90. Что такое проблема исчезающего градиента?
91. Что представляет собой архитектура LSTM?

92. Чем LSTM отличается от обычной RNN?
93. Что такое GRU?
94. Какие преимущества имеют LSTM и GRU?
95. Что представляет собой механизм внимания (attention)?
96. Для чего используется механизм внимания?
97. Что такое self-attention?
98. Что представляют собой Query, Key и Value?
99. Что такое архитектура Transformer?
100. Какие основные компоненты входят в Transformer?
101. Что такое multi-head attention?
102. Для чего используется позиционное кодирование?
103. Какие задачи решаются с помощью Transformer?
104. Что представляет собой инференс модели?
105. Чем отличается обучение модели от инференса?
106. Как измеряется производительность модели?
107. Что такое время инференса?
108. Какие методы используются для ускорения инференса?
109. Что представляет собой сохранение модели?
110. Какие способы сохранения моделей существуют в PyTorch?
111. Как загрузить сохранённую модель?
112. Что такое жизненный цикл модели машинного обучения?
113. Какие этапы включает внедрение модели в систему?
114. Что представляет собой мониторинг модели после внедрения?
115. Какие проблемы могут возникнуть при внедрении модели?
116. Что такое воспроизводимость экспериментов в машинном обучении?
117. Почему важно фиксировать случайные значения (seed)?
118. Что представляет собой конфигурация эксперимента?
119. Как организовать структуру проекта машинного обучения?
120. Какие факторы необходимо учитывать при подготовке модели к внедрению?

ОБЩИЕ РЕКОМЕНДАЦИИ ПО ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Основной формой самостоятельной работы студента является изучение конспекта лекций, их дополнение, рекомендованной литературы, активное участие на лабораторных занятиях. Но для успешной учебной деятельности, ее интенсификации, необходимо учитывать следующие субъективные факторы:

1. Знание школьного программного материала, наличие прочной системы зияний, необходимой для усвоения основных вузовских курсов. Это особенно важно для математических дисциплин. Необходимо отличать пробелы в знаниях, затрудняющие усвоение нового материала, от малых способностей. Затратив силы на преодоление этих пробелов, студент обеспечит себе нормальную успеваемость и поверит в свои способности.

2. Наличие умений, навыков умственного труда:

а) умение конспектировать на лекции и при работе с книгой;

б) владение логическими операциями: сравнение, анализ, синтез, обобщение, определение понятий, правила систематизации и классификации.

3. Специфика познавательных психических процессов: внимание, память, речь, наблюдательность, интеллект и мышление. Слабое развитие каждого из них становится серьезным препятствием в учебе.

4. Хорошая работоспособность, которая обеспечивается нормальным физическим состоянием. Ведь серьезное учение — это большой многосторонний и разнообразный труд. Результат обучения оценивается не количеством сообщаемой информации, а качеством ее

усвоения, умением ее использовать и развитием у себя способности к дальнейшему самостоятельному образованию.

5. Соответствие избранной деятельности, профессии индивидуальным способностям. Необходимо выработать у себя умение саморегулировать свое эмоциональное состояние и устранять обстоятельства, нарушающие деловой настрой, мешающие намеченной работе.

6. Овладение оптимальным стилем работы, обеспечивающим успех в деятельности. Чередование труда и пауз в работе, периоды отдыха, индивидуально обоснованная норма продолжительности сна, предпочтение вечерних или утренних занятий, стрессоустойчивость на экзаменах и особенности подготовки к ним,

7. Уровень требований к себе, определяемый сложившейся самооценкой.

Адекватная оценка знаний, достоинств, недостатков - важная составляющая самоорганизации человека, без нее невозможна успешная работа по управлению своим поведением, деятельностью.

Одна из основных особенностей обучения в высшей школе заключается в том, что постоянный внешний контроль заменяется самоконтролем, активная роль в обучении принадлежит уже не столько преподавателю, сколько студенту.

Зная основные методы научной организации умственного труда, можно при наименьших затратах времени, средств и трудовых усилий достичь наилучших результатов.

Эффективность усвоения поступающей информации зависит от работоспособности человека в тот или иной момент его деятельности.

Работоспособность - способность человека к труду с высокой степенью напряженности в течение определенного времени. Различают внутренние и внешние факторы работоспособности.

К внутренним факторам работоспособности относятся интеллектуальные особенности, воля, состояние здоровья.

К внешним:

- организация рабочего места, режим труда и отдыха;
- уровень организации труда - умение получить справку и пользоваться информацией;
- величина умственной нагрузки.

Выдающийся русский физиолог Н. Е. Введенский выделил следующие условия продуктивности умственной деятельности:

- во всякий труд нужно входить постепенно;
- мерность и ритм работы. Разным людям присущ более или менее разный темп работы;
- привычная последовательность и систематичность деятельности;
- правильное чередование труда и отдыха.

Отдых не предполагает обязательного полного бездействия со стороны человека, он может быть достигнут простой переменой дела. В течение дня работоспособность изменяется. Наиболее плодотворным является *утреннее время (с 8 до 14 часов),* причем максимальная работоспособность приходится на период с 10 до 13 часов, затем *послеобеденное* - (с 16 до 19 часов) и *вечернее* (с 20 до 24 часов). Очень трудный для понимания материал лучше изучать в начале каждого отрезка времени (лучше всего утреннего) после хорошего отдыха. Через 1-1,5 часа нужны перерывы по 10 - 15 мин, через 3 - 4 часа работы отдых должен быть продолжительным - около часа.

Составной частью научной организации умственного труда является овладение техникой умственного труда.

Время, которым располагает студент для выполнения учебного плана, складывается из двух составляющих: одна из них - это аудиторная работа в вузе по расписанию занятий,

другая - внеаудиторная самостоятельная работа. Задания и материалы для самостоятельной работы выдаются во время учебных занятий по расписанию, на этих же занятиях преподаватель осуществляет контроль за самостоятельной работой, а также оказывает помощь студентам по правильной организации работы.

Самостоятельные занятия потребуют интенсивного умственного труда, который необходимо не только правильно организовать, но и стимулировать. При этом очень важно уметь поддерживать устойчивое внимание к изучаемому материалу. Выработка внимания требует значительных волевых усилий. Именно поэтому, если студент замечает, что он часто отвлекается во время самостоятельных занятий, ему надо заставить себя сосредоточиться. Подобную процедуру необходимо проделывать постоянно, так как это является тренировкой внимания. Устойчивое внимание появляется тогда, когда человек относится к делу с интересом.

Следует правильно организовать свои занятия по времени: 50 минут - работа, 5-10 минут - перерыв; после 3 часов работы перерыв - 20-25 минут. Иначе нарастающее утомление повлечет неустойчивость внимания. Очень существенным фактором, влияющим на повышение умственной работоспособности, являются систематические занятия физической культурой. Организация активного отдыха предусматривает чередование умственной и физической деятельности, что полностью восстанавливает работоспособность человека.

УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Основная литература:

Болотский, А. В.
    Исследование операций и методы оптимизации Электронный ресурс / Болотский А. В., Кочеткова О. А. - Санкт-Петербург : Лань, 2020. - 116 с. - ISBN 978-5-8114-4568-4

Сараев, П. В.
    Методы машинного обучения Электронный ресурс : Методические указания и задания к лабораторным работам по курсу / П. В. Сараев. - Липецк : Липецкий государственный технический университет, ЭБС АСВ, 2017. - 48 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 2227-8397

Сопов, Е. А.
    Многокритериальные нейроэволюционные системы в задачах машинного обучения и человеко-машинного взаимодействия Электронный ресурс / Сопов Е. А., Иванов И. А. : монография. - Красноярск : СФУ, 2019. - 160 с. - ISBN 978-5-7638-3969-2

Дополнительная литература:

Методические рекомендации к практическим занятиям по дисциплине "Теория и практика машинного перевода" : Направление подготовки 45.04.02 Лингвистика. Магистерская программа "Перевод и переводоведение".Квалификация выпускника - магистр. Форма обучения - очная. Учебный план 2015 года. Изучается в 3 семестре. - Ставрополь : СКФУ, 2015. - 9 с.

Неделько, В. М.
    Основы статистических методов машинного обучения Электронный ресурс : Учебное пособие / В. М. Неделько. - Новосибирск : Новосибирский государственный технический университет, 2010. - 72 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 978-5-7782-1385-2

Рапаков, Г. Г.
    Методы и алгоритмы машинного обучения при принятии управленческих решений в региональной системе медицинской профилактики (опыт Вологодской области) Электронный ресурс / Рапаков Г. Г., Касимов Р. А. : монография. - Вологда : ВоГУ, 2014. - 143 с. - ISBN 978-5-87851-548-1

Учебно-методическая литература:

Сараев, П. В.
    Методы машинного обучения Электронный ресурс : Методические указания и задания к лабораторным работам по курсу / П. В.

Сараев. - Липецк : Липецкий государственный технический университет, ЭБС АСВ, 2017. - 48 с. - Книга находится в премиум-версии ЭБС IPR BOOKS. - ISBN 2227-8397