Общество с ограниченной ответственностью

«Проектная школа - Самара»

****

443015 г. Самара, ул. Главная, д.3, оф.20, тел. 8(927)777-09-67

E-mail: mironov@100lines.ru

УТВЕРЖДАЮ

Директор

ООО «Проектная школа — Самара»

_____ Е.Н.Миронов

15 декабря 2021 г.

ДОПОЛНИТЕЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ

(ОБЩЕРАЗВИВАЮЩАЯ) ПРОГРАММА

«СОЗДАНИЕ НЕЙРОННЫХ СЕТЕЙ НА PYTHON»

Возраст детей: от 13 лет

Срок обучения: 72 академических часа

Автор-составитель:

Мальцев С.А., педагог ДО

г.о. Самара, 2021

Введение

Программа «Создание нейронных сетей на Python» разработана в соответствии с требованиями к дополнительным общеобразовательным общеразвивающим программам технической направленности и направлена на формирование у обучающихся навыков проектирования, обучения и применения искусственных нейронных сетей для решения задач компьютерного зрения, обработки естественного языка и генеративного моделирования.

Искусственный интеллект и машинное обучение являются одними из самых быстрорастущих областей современной IT-индустрии. Python является основным языком в этой сфере благодаря мощным библиотекам таким как NumPy, TensorFlow, Keras и PyTorch. Программа обеспечивает переход от теоретического понимания математических основ нейросетей к практической реализации сложных архитектур глубокого обучения.

Актуальность программы обусловлена высоким спросом на специалистов в области Data Science и Machine Learning. Курс позволяет обучающимся не просто использовать готовые API, но и понимать внутреннюю работу алгоритмов, что критически важно для создания эффективных и оптимизированных моделей.

1. Пояснительная записка

Актуальность программы. В эпоху цифровой трансформации технологии машинного обучения проникают во все сферы жизни: от медицины и финансов до развлечений и искусства. Знание принципов работы нейронных сетей становится конкурентным преимуществом. Программа актуальна, так как дает практические навыки работы с двумя ведущими фреймворками глубокого обучения (TensorFlow/Keras и PyTorch), позволяя обучающимся создавать реальные продукты: системы распознавания изображений, чат-боты, генераторы музыки и арта.

Отличительные особенности программы. Программа сочетает глубокое погружение в математику процессов (градиентный спуск, функции потерь, обратное распространение ошибки) с интенсивной практикой. Обучающиеся работают с реальными датасетами (MNIST, CIFAR-10, Fashion-MNIST), учатся бороться с переобучением, настраивать гиперпараметры и участвовать в соревнованиях по машинному обучению (Kaggle). Особое внимание уделяется сравнению подходов разных фреймворков и выбору оптимального инструмента под конкретную задачу.

Учебные дисциплины:

Линейная алгебра и основы математического анализа для ML. Архитектура искусственного нейрона и многослойного перцептрона. Глубокое обучение (Deep Learning): сверточные сети (CNN), рекуррентные сети (RNN/LSTM), автоэнкодеры, генеративно-состязательные сети (GAN). Работа с фреймворками TensorFlow, Keras, PyTorch. Предобработка данных и feature engineering.

Практические дисциплины:

Реализация нейросети «с нуля» на NumPy. Классификация изображений и текста. Генерация текстов, аудио и изображений. Сегментация изображений. Участие в учебном соревновании по анализу данных.

Цель образовательной программы: формирование у обучающихся компетенций в области глубокого обучения, способности самостоятельно проектировать, обучать и оценивать качество нейронных сетей для решения прикладных задач.

Задачи программы:

Обучающие:

- познакомить с математическими основами работы нейронных сетей (линейная алгебра, градиентный спуск);
- научить реализовывать простейший нейрон и многослойный перцептрон без использования высокоуровневых библиотек;
- освоить работу с библиотеками TensorFlow, Keras и PyTorch;
- изучить архитектуры CNN для задач компьютерного зрения и RNN/LSTM для последовательностей;
- научить применять методы регуляризации для борьбы с переобучением;
- освоить продвинутые архитектуры: Autoencoders и GANs для генерации данных;
- научить проводить предобработку данных различных типов (изображения, текст, аудио).

Воспитывающие:

- воспитывать культуру экспериментальной работы: фиксацию результатов, анализ ошибок, ведение логов обучения;
- формировать этическое отношение к использованию ИИ (проблемы bias в данных, deepfakes);
- воспитывать настойчивость в поиске оптимальных гиперпараметров и архитектур.

Развивающие:

- развивать аналитическое мышление и способность интерпретировать графики обучения (loss/accuracy curves);
- развивать навыки исследовательской деятельности через участие в симуляции Kaggle-соревнований;
- развивать навыки презентации технических решений и защиты проектов.

2. Принципы, формы и методы обучения

Основными **принципами обучения** являются:

1. Научность. Изучаются фундаментальные концепции машинного обучения и глубокого обучения (матричные вычисления, градиентный спуск, архитектуры нейросетей), что создает прочную теоретическую базу для дальнейшего профессионального роста.
2. Доступность. Сложные математические абстракции объясняются через практические задачи, визуализацию процессов и пошаговое написание кода. Python обладает мощными библиотеками, снижающими порог входа в Data Science.
3. Связь теории с практикой. Каждая изученная архитектурная концепция (CNN, RNN, GAN) немедленно закрепляется решением прикладной задачи на реальных датасетах.
4. Воспитательный характер обучения. Формируется культура научного эксперимента: фиксация гиперпараметров, анализ метрик, честность в оценке результатов, понимание этических аспектов ИИ.
5. Активность обучения. Обучающиеся не просто запускают готовый код, а самостоятельно проектируют архитектуру сети, выбирают функции потерь и интерпретируют причины успеха или неудачи модели.
6. Наглядность. Использование интерактивных сред (Jupyter Notebook), визуализация графиков обучения, демонстрация работы генеративных моделей.
7. Систематичность и последовательность. Материал подается от простого к сложному: от линейной алгебры и одного нейрона — к сверточным сетям и генеративным моделям.
8. Индивидуальный подход. Педагог учитывает уровень математической подготовки обучающихся, предлагая дополнительные материалы или углубленные задачи по оптимизации.

Формы организации образовательного процесса: сочетание фронтально-групповой и индивидуальной форм (групповое обсуждение идей и архитектур, индивидуальное обучение моделей, коллективное тестирование и разбор ошибок).

Методы обучения:

1. Объяснительно-иллюстративный — демонстрация работы алгоритмов, разбор архитектур популярных нейросетей, работа с документацией фреймворков.
2. Эвристический — самостоятельный подбор архитектуры и гиперпараметров для достижения лучших метрик.
3. Проблемный — постановка задачи («почему модель переобучается?») и поиск решения через анализ графиков.
4. Программированный — выполнение лабораторных работ по пошаговым инструкциям в Jupyter Notebook.
5. Репродуктивный — написание кода по образцу с последующей модификацией под свои данные.
6. Поисковый — решение задач с использованием официальной документации TensorFlow/PyTorch и научных статей.
7. Метод проектов — основной метод, включающий этапы: сбор данных → выбор архитектуры → обучение → оценка качества → защита проекта.

3. Механизм отслеживания результатов

Контроль освоения учебного материала проходит в три этапа:

1. Входной мониторинг: тест на знание основ Python, линейной алгебры и базового понимания статистики.
2. Промежуточный контроль: защита практических работ (реализация нейрона на NumPy, первая модель на Keras, первая модель на PyTorch).
3. Итоговый мониторинг: участие в итоговом соревновании (Kaggle-like) и защита финального проекта (генеративная модель или сложная классификационная задача).

В конце курса проводится анализ качества данной программы (содержания, методической подачи и организационных моментов) и, по необходимости, проводится коррекция программы для следующего набора.

4. Прогнозируемый результат

В конце обучения по данному курсу ученики должны знать:

- математические основы нейронных сетей (матричные операции, производные, градиентный спуск);
- архитектуру прямого распространения (Feedforward) и обратного распространения ошибки (Backpropagation);
- различия между функциями активации (Sigmoid, ReLU, Softmax, Tanh) и их применение;

- принципы работы сверточных нейронных сетей (CNN) и рекуррентных сетей (RNN/LSTM);
- понятия переобучения (overfitting) и недообучения (underfitting), методы регуляризации (Dropout, L1/L2);
- основы работы с фреймворками TensorFlow/Keras и PyTorch;
- принципы работы генеративно-состязательных сетей (GAN) и автоэнкодеров;
- метрики оценки качества моделей (Accuracy, Precision, Recall, F1-score, Loss).

В конце обучения по данному курсу ученики должны уметь:

- предобрабатывать данные (изображения, текст, аудио) для подачи в нейросеть;
- строить, компилировать и обучать нейронные сети в Keras и PyTorch;
- анализировать графики обучения и диагностировать проблемы модели;
- настраивать гиперпараметры (learning rate, batch size, epochs) для улучшения результатов;
- реализовывать задачи классификации изображений и текста;
- создавать генеративные модели для синтеза текста, звука и изображений;
- использовать техники Transfer Learning (переноса обучения);
- оформлять результаты экспериментов и представлять их в виде отчетов или презентаций.

5. Учебно-тематический план обучения

№ п/п	Тема	часы		
		всего	теория	практ.
1	Тема 1. Введение в профессию Data Scientist и ML Engineer. Обзор рынка и инструментов.	2	1,0	1,0
2	Тема 2. Продолжение введения. Установка окружения (Anaconda, Jupyter, GPU drivers). Основы линейной алгебры для ML.	2	1,0	1,0
3	Тема 3. Введение в NumPy. Векторизация вычислений. Операции с матрицами и тензорами.	2	0,5	1,5
4	Тема 4. Первый нейрон. Реализация перцептрона с нуля на Python/NumPy. Функция активации.	2	0,5	1,5
5	Тема 5. Продолжаем изучение нейронов. Многослойный перцептрон (MLP). Прямое распространение сигнала.	2	0,5	1,5
6	Тема 6. Продолжаем изучение нейронов. Обратное	2	0,5	1,5

	распространение ошибки (Backpropagation). Расчет градиентов.			
7	Тема 7. Знакомство с Tensorflow. Установка, структура графа вычислений, Eager Execution.	2	0,5	1,5
8	Тема 8. Градиентный спуск. Виды оптимизаторов (SGD, Adam, RMSprop). Learning Rate.	2	0,5	1,5
9	Тема 9. Скрытый слой, нелинейность. Зачем нужны скрытые слои? Влияние нелинейных функций активации.	2	0,5	1,5
10	Тема 10. MNIST. Первая полноценная нейросеть для распознавания рукописных цифр. Оценка качества.	2	0,2	1,8
11	Тема 11. Регуляризация. Переобучение. Методы борьбы: Dropout, L1/L2 регуляризация, Early Stopping.	2	0,5	1,5
12	Тема 12. Функции активации. Подробный разбор Sigmoid, Tanh, ReLU, Leaky ReLU, Softmax. Выбор функции.	2	0,5	1,5
13	Тема 13. Подробное знакомство с Keras. Sequential и Functional API. Повторение задачи MNIST на Keras.	2	0,5	1,5
14	Тема 14. Keras - классификация изображений. Сверточный слой (Conv2D). Пулинг. Датасет Fashion MNIST.	2	0,5	1,5
15	Тема 15. Keras - Cifar10. Углубление в CNN. Batch Normalization. Работа с цветными изображениями.	2	0,2	1,8
16	Тема 16. Keras - Генерация текста. Рекуррентные нейросети (RNN). Подготовка текстовых данных.	2	0,5	1,5
17	Тема 17. Keras - Генерация текста - 2. LSTM и GRU ячейки. Обучение модели на корпусе текстов.	2	0,2	1,8
18	Тема 18. Keras - Классификация аудио. Преобразование звука в спектрограммы. CNN для аудио.	2	0,5	1,5
19	Тема 19. Pytorch. Введение. Тензоры PyTorch. Отличие от TensorFlow. Динамические графы.	2	0,5	1,5
20	Тема 20. Pytorch. Классификация. Создание Dataset и DataLoader. Обучение простой сети на PyTorch.	2	0,2	1,8
21	Тема 21. Pytorch. Анализ отзывов. NLP задачи в PyTorch.	2	0,5	1,5

	Embedding слои. Классификация тональности.			
22	Тема 22. Pytorch. Генерация изображений. Автоэнкодер (Autoencoder). Архитектура Encoder-Decoder.	2	0,5	1,5
23	Тема 23. Pytorch. Генерация изображений. GAN (Generative Adversarial Networks). Теория игры Generator vs Discriminator.	2	0,5	1,5
24	Тема 24. Pytorch. Генерация изображений. GAN. 2. Практическая реализация DCGAN. Обучение и проблемы сходимости.	2	0,2	1,8
25	Тема 25. Keras. Решаем задачи. Сборник практических задач на закрепление материала CNN и RNN.	2	0,0	2,0
26	Тема 26. Keras. GAN. Реализация генеративно-состязательной сети в экосистеме Keras/TensorFlow.	2	0,5	1,5
27	Тема 27. Keras. GAN. Генерация лиц. Работа с датасетом CelebA или Faces. Улучшение качества генерации.	2	0,2	1,8
28	Тема 28. Keras. Классификация кошек и собак. Transfer Learning. Использование предобученных моделей (VGG, ResNet).	2	0,5	1,5
29	Тема 29. Keras. Сегментация изображений. U-Net архитектура. Пиксельная классификация.	2	0,5	1,5
30	Тема 30. Keras. Напишем песню. Генерация MIDI-последовательностей или аудио с помощью нейросетей.	2	0,2	1,8
31	Тема 31. Keras. Классификация. Решение сложной задачи классификации с аугментацией данных.	2	0,2	1,8
32	Тема 32. Keras. Классификация. Оптимизация архитектуры и гиперпараметров для задачи из Темы 31.	2	0,0	2,0
33	Тема 33. Keras. Классификация. Финальная доводка модели, ансамблирование моделей.	2	0,0	2,0
34	Тема 34. Соревнование. Внутренний хакатон: решение задачи от преподавателя на время и точность.	2	0,0	2,0
35	Тема 35. Keras. Классификация. Разбор ошибок соревнования, анализ лучших решений участников.	2	0,5	1,5
36	Тема 36. Kaggle и гиперпараметры. Обзор платформы Kaggle. Автоматический поиск гиперпараметров. Итоги	2	1,0	1,0

	курса.			
	ИТОГО	**72**	**15,1**	**56,9**

6. Обеспечение программы

Для организации образовательного процесса по программе необходимы следующие ресурсы.

Кадровые: Специалист, имеющий высшее образование (педагогическое, техническое или IT-направление), владеющий языком Python, фреймворками TensorFlow и PyTorch, знаниями и методикой преподавания курса, понимающий возрастную психологию подростков.

Материально-технические: Компьютерный класс, оснащенный персональными компьютерами или ноутбуками (1 место на 1 учащегося) с дискретными видеокартами (поддержка CUDA) или стабильным доступом к облачным платформам; настенная доска (маркерная или интерактивная); мультимедийный проектор и экран; система стабилизированного электроснабжения; доступ к сети Интернет. Помещение должно соответствовать санитарно-эпидемиологическим требованиям (СанПиН) к организации работы с ЭВМ.

Программно-методические:

1. Интерпретатор Python 3.8+, установленный на каждый ПК (или настроенные аккаунты в облачных средах).
2. Интегрированная среда разработки (Jupyter Notebook, VS Code или PyCharm Professional).
3. Веб-браузеры с настроенной системой фильтрации контента для доступа к официальным ресурсам и репозиториям датасетов.
4. Методическое пособие (поурочные планы, конспекты лекций по математике ML, описание лабораторных работ).
5. Раздаточный материал (шпаргалки по архитектуре сетей, чек-листы предобработки данных, шаблоны Jupyter-ноутбуков).
6. Мультимедийные презентации к каждому занятию.

7. Литература

Нормативно-правовая база:

1. Федеральный закон РФ «Об образовании в Российской Федерации» от 29.12.2012 № 273-ФЗ.
2. СанПиН 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи».
3. Концепция развития дополнительного образования детей до 2030 года.

Для педагога (учебная и методическая литература):

4. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение. — М.: ДМК Пресс, 2018.
5. Шолле Ф. Глубокое обучение на Python. — СПб.: Питер, 2021.
6. Пасканов А. Нейронные сети. Полный курс. — М.: Эксмо, 2022.
7. Расчка В. Python и машинное обучение. — СПб.: БХВ-Петербург, 2023.

Для обучающихся:

8. Бурков Е. Элементы глубокого обучения. — М.: ДМК Пресс, 2021.
9. Хастие Т., Тибширани Р., Фридман Дж. Введение в статистическое обучение. — М.: Вильямс, 2022.
10. Свейгарт Э. Автоматизация рутинных задач с помощью Python. — М.: Вильямс, 2022.

Электронные ресурсы:

11. Официальная документация TensorFlow: https://www.tensorflow.org
12. Официальная документация PyTorch: https://pytorch.org
13. Kaggle Learn — бесплатные микро-курсы: https://www.kaggle.com/learn
14. Metanit — учебник по Python и ML: https://metanit.com 15. GitHub — платформа для размещения проектов: https://github.com