МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Систем обработки информации и управления»

ОТЧЕТ

**Лабораторная работа №\_\_1\_\_**

по дисциплине«Проектирование нейросетевых систем»

Тема: «Введение в DL»

ИСПОЛНИТЕЛЬ: \_\_\_Винников С.С.\_\_\_

ФИО

группа ИУ5-24М \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"20"\_февраля\_2024 г.

ПРЕПОДАВАТЕЛЬ: \_\_\_Канев А.И.\_\_\_\_\_

ФИО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"\_\_"\_\_\_\_\_\_\_\_\_2024г.

Москва - 2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Задание

Необходимо познакомиться с фреймворком машинного обучения PyTorch и выполнить три задания:

1. Регрессия по теореме универсальной аппроксимации, ручное дифференцирование
2. Бинарная классификация с помощью автодиффиренцирования PyTorch
3. Обучить полносвязную нейронную сеть классификации 3 классов изображений из набора данных CIFAR100 по варианту из примера и затем повысить точность на тестовой выборке.

Для задания 3 нужно сформировать свою подвыборку CIFAR100 по варианту. Вы должны использовать следующие классы из CIFAR100:

1. Номер группы + 15
2. Номер варианта + 56
3. ИУ5 (Номер варианта + 21); ГУИМЦ (80); Иностранцы (90)

Полученные варианты:

1. 24 + 15 = 39
2. 2 + 56 = 58
3. 2 + 21 = 23

# Часть 1. Задача регрессии по теореме универсальной аппроксимации, ручное дифференцирование

Выборка:

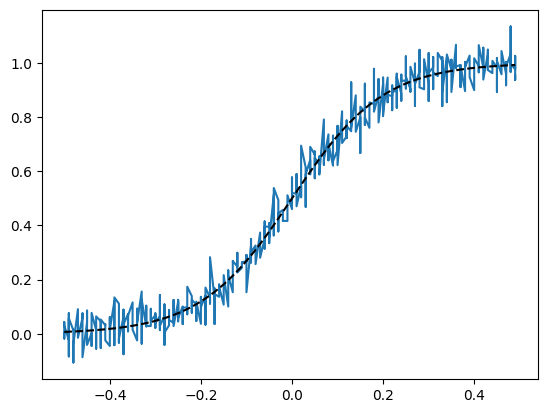


Рис. 1 – аппроксимируемая функция.

Решается задача аппроксимации функции – отмечена на рис.1 пунктиром. В значения “y” был внесён сдвиг со стандартным квадратическим отклонением, равным 0.05

Аппроксимация производится с помощью градиентного спуска. Функция потерь – сумма квадратов отклонения. Параметры НС: 1 вход, 1 скрытый слой из 64 нейронов, 1 выход.

На каждой итерации высчитывается производная по весам и смещению для обоих уровней НС, после чего значения сдвигаются на x\_new = x\_old - 0,0001\*grad(x).

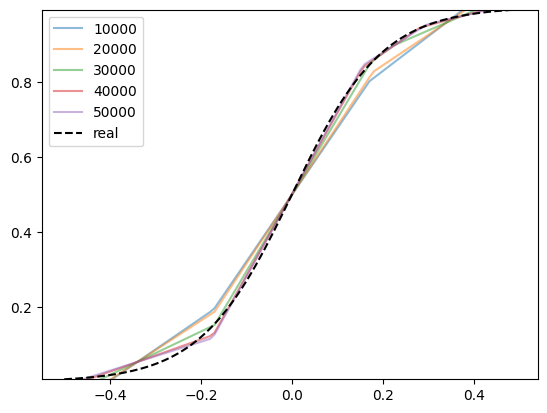


Рис. 2 – результат аппроксимации.

При прохождении 50000 итераций сумма квадратов потерь теперь составляет 0.0025866674. Результат аппроксимации указан на рис. 2.

# Часть 2. Бинарная классификация с помощью автодиффиренцирования PyTorch

Решается задача классификации. Классы отображены на рис.3. Параметры НС: 2 входа (соответственно x и y), 1 скрытый слой из 16 нейронов, 1 выход – класс.

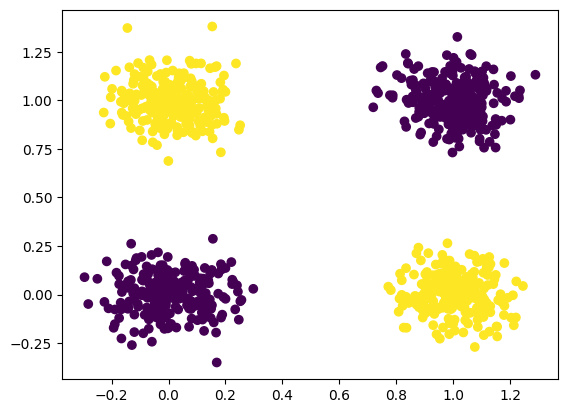


Рис. 3 – график классов.

Функция потерь – логарифмическая

чем дальше значение от , тем ближе значение к 1. Соответственно для всего объёма значений берём сумму всех потерь.

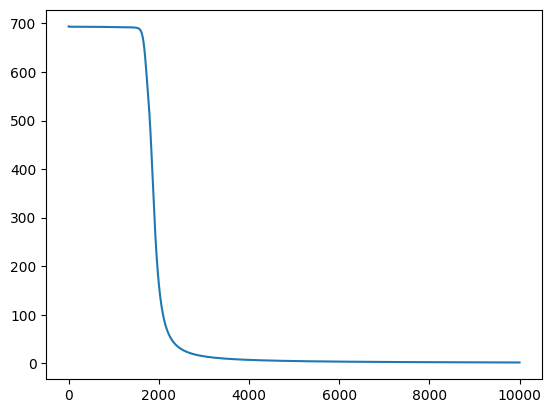


Рис. 4 – график общей функции потерь.

По рисунку видно, что после около 2.7 тысяч итераций НС начала сильно уменьшать функцию потерь, т.е. было найдено верное направление градиента. После 8000 итераций функция потерь находится у минимума. На рис.5 изображён график вывода нейронной сети в области [-0.5;0.5]

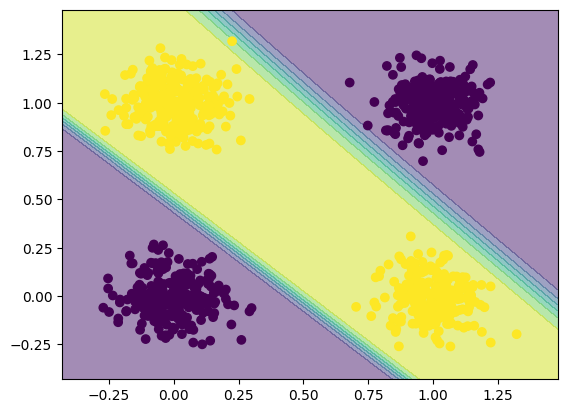


Рис. 5 – полученные области классификации.

Поскольку вначале графика потерь видим плато, можно говорить о потенциальном сокращении числа итераций до нахождения минимума, путём увеличения learning rate.

|  |  |  |
| --- | --- | --- |
| **C:\Users\ksarb\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\A21FE299.tmp** | C:\Users\ksarb\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\DB7334F5.tmp |  |
| Рис. 6 – функция ошибок с lr 1e-2 | Рис. 7 – функция ошибок с lr 2e-3 |  |

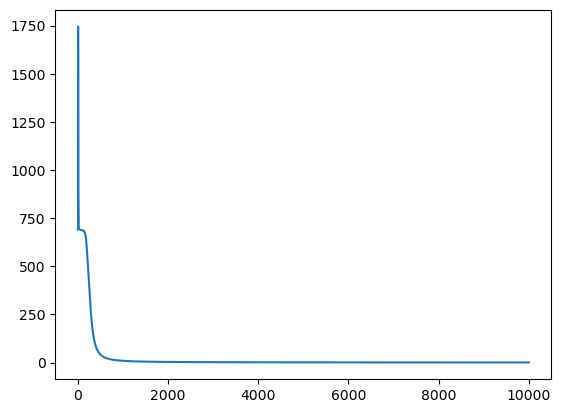


Рис. 8 – функция ошибок с lr 3e-3

Для learning rate 0,003 (рис. 8) видим почти мгновенное начало уменьшения функции ошибок, причём не попадаем в локальный минимум, наблюдаемый при lr1e-2 (рис.6), и не наблюдая плато, как на рис.7.

Проверка для lr 3e-3:

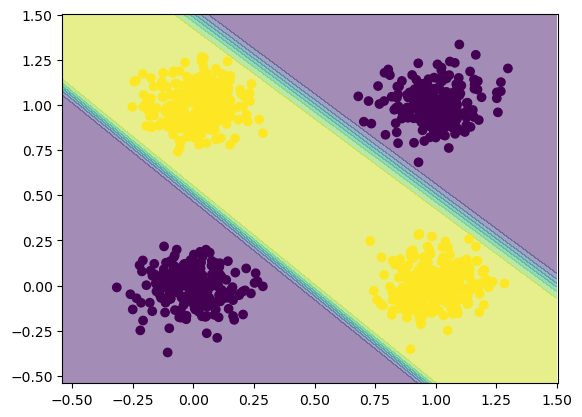


Рис. 9 – полученные области для lr 3e-3.

# Часть 3. Классификация изображений CIFAR100

Вариант: CLASSES = [39, 58, 23], где класс 39 — клавиатуры, 58 — внедорожники, 23 — облака.

Взглянем ближе на классы изображений, которые должна распознать модель. Ниже представлены рисунки изображений каждого класса:



Рис. 12 – класс с клавиатурами



Рис. 13 – класс с внедорожниками



Рис. 14 – класс с облаками

Для базового варианта нейронной сети, получаем показатели метрик, представленные в таблице 1.

Таблица 1. Показатели метрик для базовой модели

|  |  |  |  |
| --- | --- | --- | --- |
| **Train** | | | |
| Классы | **Precision** | **Recall** | **F1-score** |
| Клавиатуры | 0.9878 | 0.9700 | 0.9788 |
| Внедорожники | 0.9763 | 0.9880 | 0.9821 |
| Облака | 0.9881 | 0.9940 | 0.9910 |
| **Test** | | | |
| Классы | **Precision** | **Recall** | **F1-score** |
| Клавиатуры | 0.6583 | 0.7900 | 0.7182 |
| Внедорожники | 0.6207 | 0.5400 | 0.5775 |
| Облака | 0.7957 | 0.7400 | 0.7668 |

Видим, что показатели обучающей выборки довольны высоки, однако для тестовой выборки показатели примерно равны 75%. Скорее всего происходит переобучение НС.

Проверим происходит ли переобучение модели. Действительно, минимальная ошибка на тестовых данных достигается на 75 эпохе, со значением 0.619.

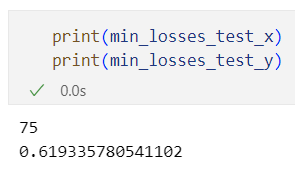


Рис. 15 – минимальное значение val loss для базовой модели.

Попробуем обучить модель до этой эпохи (75) и сравним результаты. Результаты обучения представлены в таблице 2.

Таблица 2. Показатели метрик для базовой модели с уменьшенным количеством эпох.

|  |  |  |  |
| --- | --- | --- | --- |
| **Train** | | | |
| Классы | **Precision** | **Recall** | **F1-score** |
| Клавиатуры | 0.9010 | 0.9280 | 0.9143 |
| Внедорожники | 0.9367 | 0.8880 | 0.9117 |
| Облака | 0.9511 | 0.9720 | 0.9614 |
| **Test** | | | |
| Классы | **Precision** | **Recall** | **F1-score** |
| Клавиатуры | 0.6783 | 0.7800 | 0.7256 |
| Внедорожники | 0.6905 | 0.5800 | 0.6304 |
| Облака | 0.8416 | 0.8500 | 0.8458 |

Результаты обучающей выборки ухудшились, как и следовало ожидать, так как теперь мы приблизились к моменту переобучения. Однако, улучшились результаты на тестовой выборке.

Локальные максимумы указаны в таб.2.

Таблица 2 – итоговая таблица.

|  |  |  |  |
| --- | --- | --- | --- |
| **Конфигурация нейросети** | **Гиперпараметры** | **Точность** | **Комментарий** |
| FC(10), FC(3) | lr = 0.005, batch\_size = 128, epochs = 250 | test = 69%, train = 98% | Базовый вариант |
| FC(10), FC(3) | lr = 0.005, batch\_size = 128, epochs = 75 | test = 73%, train = 92% | В базовой модели было переобучение – уменьшили количество эпох. |
| FC(10), FC(3) | lr = 0.001, batch\_size = 128, epochs = 375 | test = 72%, train = 93% | Уменьшили learning rate для повышения точности. Соответственно увеличили количество эпох |
| FC(10), FC(3) | lr = 0.001, batch\_size = 128, epochs = 314 | test = 75%, train = 92% | Обнаружено переобучение для прошлой итерации. Снизили количество эпох до 314. |
| FC(10), FC(3) | lr = 0.001, batch\_size = 192, epochs = 471 | test = 77%, train = 91% | Увеличение батч сайз в 1.5 раза |
| FC(10), FC(3) | lr = 0.001, batch\_size = 192, epochs = 417 | test = 76%, train = 90% | Обнаружено переобучение для прошлой итерации. Снизили количество эпох до 417. |
| FC(8), FC(3) | lr = 0.005, batch\_size = 128, epochs = 250 | test = 73%, train = 97% | Создали новую модель FC(8), FC(3). |
| FC(8), FC(3) | lr = 0.001, batch\_size = 128, epochs = 310 | test = 74%, train = 85% | Приведение к оптимальным гиперпараметрам. |
| FC(8), FC(8), FC(3) | lr = 0.005, batch\_size = 128, epochs = 250 | test = 69%, train = 97% | Создали новую модель FC(8), FC(8), FC(3). |
| FC(8), FC(8), FC(3) | lr = 0.001, batch\_size = 128, epochs =450 | test = 72%, train = 83% | Приведение к оптимальным гиперпараметрам. |

**Вывод:** при обучении нейронных сетей очень важны гиперпараметры. Очень важно грамотно определить момент наступления переобучения, и по возможности уменьшить learning rate – это приводит к увеличению качества НС.



Замечания:

1. Нет вывода
2. Таблица не оформлена
3. Не поменял гиперпараметры. Нужно сделать правильно и правильные цифры.
4. Сделать третью модель.
5. Проанализировать recall, precioson по классам и с выводом.
6. Рисунок сделать таблицей.