Лабораторная работа №3
Многослойная искусственная нейронная сеть

Цель работы: изучить обучение и функционирование многослойной ИНС при решении задач прогнозирования.

Материал и формулы были подробно рассмотрены на лекции. Если вы по каким-то причинам их не имеете, начиная с 65-й страницы книги по ссылке можете найти всю необходимую вам информацию: https://elib.bsu.by/bitstream/123456789/193558/1/Golovko.pdf

Варианты

Таблица №1: Методы предварительной обработки изображений

№ варианта	Методы предварительной обработки
1	Масштабирование (Scaling) + Поворот (Rotation)
2	Кадрирование (Cropping) + Отражение (Flipping)
3	Сдвиг (Translation) + Изменение яркости (Brightness Adjustment)
4	Поворот (Rotation) + Шум (Noise Injection)
5	Масштабирование (Scaling) + Кадрирование (Cropping)
6	Изменение яркости (Brightness Adjustment) + Изменение контраста (Contrast Adjustment)
7	Шум (Noise Injection) + Размытие (Blurring)
8	Поворот (Rotation) + Сдвиг (Translation)
9	Отражение (Flipping) + Кадрирование (Cropping)
10	Изменение контраста (Contrast Adjustment) + Шум (Noise Injection)
11	Масштабирование (Scaling) + Отражение (Flipping)

Ход работы

В Таблице предварительной обработки изображений представлены методы, которые необходимо будет согласно вашему варианту применить для каждого из изображений, подаваемого на обучающую выборку перед непосредственной подачей (чтобы изменения применялись динамически)

Для языка разработки Python существует удобная библиотека albumentations: https://github.com/albumentations-team/albumentations

Для демонстрации работы методов предварительной обработки можно использовать matplotlib.

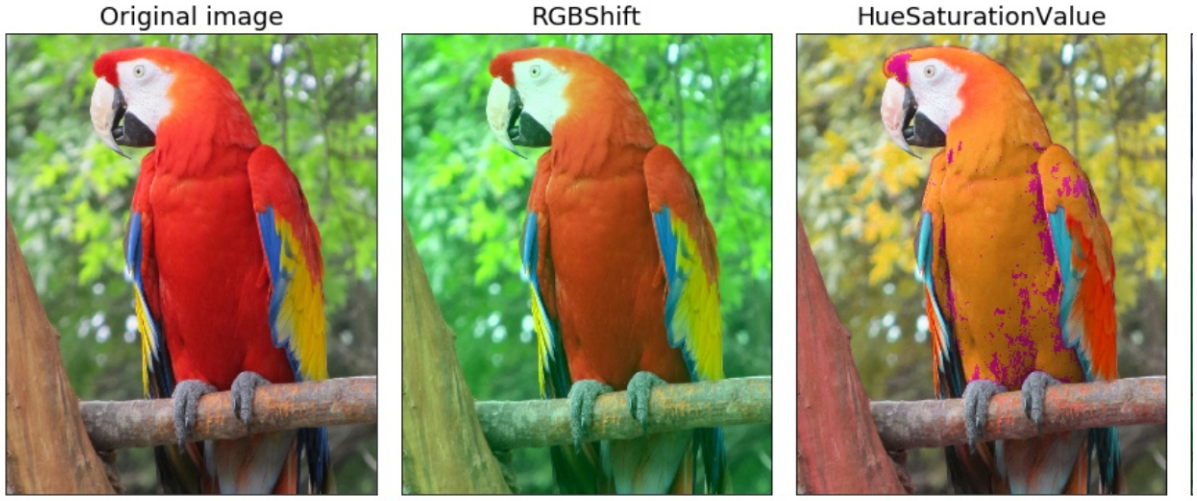
1. Скачиваем набор данных MNIST с сайта Kaggle: https://www.kaggle.com/datasets/hojjatk/mnist-dataset
2. Распределяем данные по train/test наборам
3. Выполняем предобработку данных:
    a. Согласно заданному варианту, применяем перечисленные трансформации изображений к обучающей выборке (изображения для тестирования не подвергаются изменениям)
    b. Выполняем нормирование каждого изображения в пределах от 0..1 или -1..1
    c. Переходим от матричного представления данных к векторному
4. Обучаем и тестируем нейронную сеть по эпохам до тех пор, пока итоговая точность сети будет > 0.98.

5. Сохраняем обученную нейронную сеть + реализуем метод загрузки её весов и порогов для тестирования на лабораторной работе

Примечание: допускается применять любые функции активации, произвольно выбирать число нейронов на каждом (если будет более 1-го) скрытом слое нейронной сети. Архитектура модели реализуется исключительно руками студента. По-желанию могут быть добавлены методы групповой обработки – batch-es. Информация по тому, как это сделать есть в книге В.А. Головко, что упоминалась выше.

Ожидаемый вывод работы программы

You example of your preprocessing method (with result as combo of 2 methods, like RGBShift + HueSaturationValue)



Что должна выводить консоль:

NN architecture + hyperparams

Architecture: 784 – 256 (Tanh) – 10 # example if used activation function, if no any activation will be (None)

Learning Rate: 1e-4 # example

Batch size: 1 # for classic usage by our lecture material, if no – change on your custom value

Max train epoch: 25 # example

Train & Test process

Epoch #1: train_accuracy = … ; test_accuracy = … ; time= … ; # here time is total epoch spent time for train & test
Epoch #2: -||-

...
Epoch #K: train_accuracy = 0.985; test_accuracy = 0.981