**Лабораторная работа № 1. Обучение классификаторов средствами библиотеки PyTorch**

|  |
| --- |
| **Цель:** научиться конструировать нейросетевые классификаторы и выполнять их обучение на известных выборках компьютерного зрения |

**Общее задание**

1. Выполнить конструирование своей модели СНС, обучить ее на выборке по заданию (использовать **torchvision.datasets**). Предпочтение отдавать как можно более простым архитектурам, базирующимся на базовых типах слоев (сверточный, полносвязный, подвыборочный, слой нелинейного преобразования). Оценить эффективность обучения на тестовой выборке, построить график изменения ошибки (matplotlib);

2. Ознакомьтесь с state-of-the-art результатами для предлагаемых выборок (<https://paperswithcode.com/task/image-classification>). Сделать выводы о результатах обучения СНС из п. 1;

3. Реализовать визуализацию работы СНС из пункта 1 (выбор и подачу на архитектуру произвольного изображения с выводом результата);

4. Оформить отчет по выполненной работе, загрузить исходный код и отчет в соответствующий репозиторий на github.

**Задание по вариантам**

| **№ варианта** | **Выборка** | **Размер исходного изображения** | **Оптимизатор** |
| --- | --- | --- | --- |
| 1 | MNIST | 28Х28 | SGD |
| 2 | Fashion-MNIST | 28Х28 | SGD |
| 3 | CIFAR-10 | 32X32 | SGD |
| 4 | CIFAR-100 | 32X32 | SGD |
| 5 | STL-10 (размеченная часть) | 96X96 | SGD |
| 6 | MNIST | 28Х28 | Adam |
| 7 | Fashion-MNIST | 28Х28 | Adam |
| 8 | CIFAR-10 | 32X32 | Adam |
| 9 | CIFAR-100 | 32X32 | Adam |
| 10 | STL-10 (размеченная часть) | 96X96 | Adam |
| 11 | MNIST | 28Х28 | Adadelta |
| 12 | Fashion-MNIST | 28Х28 | Adadelta |
| 13 | CIFAR-10 | 32X32 | Adadelta |
| 14 | CIFAR-100 | 32X32 | Adadelta |
| 15 | STL-10 (размеченная часть) | 96X96 | Adadelta |
| 16 | MNIST | 28Х28 | RMSprop |
| 17 | Fashion-MNIST | 28Х28 | RMSprop |
| 18 | CIFAR-10 | 32X32 | RMSprop |
| 19 | CIFAR-100 | 32X32 | RMSprop |
| 20 | STL-10 (размеченная часть) | 96X96 | RMSprop |

**Важные замечания**

1. Критерии обучения можно подобрать самостоятельно, рекомендуется использовать **CrossEntropyLoss**;

2. Гайд по MNIST: <https://nextjournal.com/gkoehler/pytorch-mnist>