**Лабораторная работа по теме «Проблема конфиденциальности данных при использовании ML»**

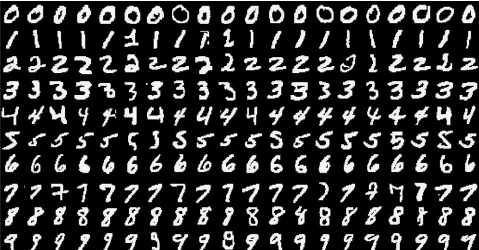
**Цель лабораторно работы**: научиться применять модель дифференциальной конфиденциальности на примере алгоритма PATE, реализовать программу, позволяющую обучить модель студента на данных датасета MNIST с сохранением конфиденциальности.

**Требования**

* ОС Windows (либо ОС Linux) с установленным Python-интерпретатором и IDE (IDE опциональна, если вы собираетесь использовать Google Colab или Jupyter Notebook);
* Подключение к сети Интернет;
* Библиотеки для Python: numpy, torch, sklearn и matplotlib.

**Постановка задачи**

Требуется реализовать алгоритм PATE, разобранный на лекции. Предполагается использование языка программирования Python для решения поставленной задачи. В качестве конфиденциального датасета используется датасет MNIST, содержащий образцы рукописных цифр пользователей в оттенках серого. Размер каждого образца 28х28 пикселей. Пример приведен ниже.



Задача состоит в обучении модели студента распознавать рукописные цифры (то есть проводить классификацию изображений на 10 классов: цифры от 0 до 9).

Данный датасет разделяется на две части. Первая используется для обучения учителей. Вторую часть считаем неразмеченной.

Пример выполнения PATE анализа:

from syft.frameworks.torch.differential\_privacy import pate

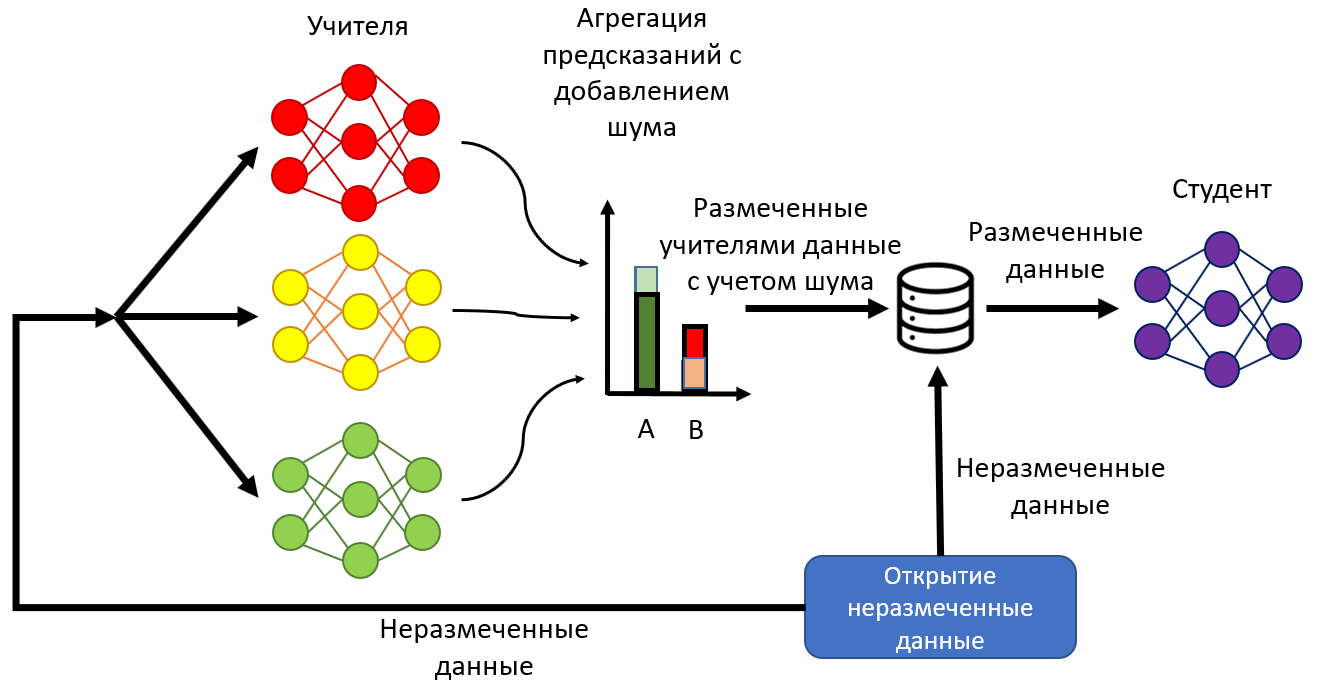
data\_dep\_eps, data\_ind\_eps = pate.perform\_analysis(teacher\_preds=teacher\_preds, indices=indices, noise\_eps=2.5, delta=1e-6, moments=140)

print("Data Independent Epsilon:", data\_ind\_eps)

print("Data Dependent Epsilon:", data\_dep\_eps)

**Рекомендации по использованию инструментов для решения задачи**

Вспомним схему обучения, которую предполагает алгоритм PATE



С помощью алгоритма PATE мы будем использовать агрегированные результаты моделей учителей для обучения другой модели - студента, на немаркированных (неполных) общедоступных данных. При этом каждая из моделей учителей учится на локальных конфиденциальных данных.

Основное предложение – использовать Google Colab.

Для того, чтобы не писать агрегатор вручную, будет использовать готовую реализацию из репозитория tensorflow. Необходимо скачать файл aggregation.py по ссылке <https://github.com/tensorflow/privacy/blob/master/research/pate_2017/aggregation.py> и не забыть импортировать в собственный проект.

Определимся с некоторыми исходными параметрами:

* Количество учителей: 200
* Количество эпох обучения учителей: 30
* Количество эпох обучения модели студента: 15
* Значение параметра приватности нужно подобрать экспериментально
* Минимальная точность распознавания модели студента на тестовой выборке не менее 80%
* Функция активации для моделей студента и учителей – Релу.
* Функция потерь NLLLoss

Исходный датасет разделяется на несколько частей. Первая используется в неизменном виде для обучения и тестирования моделей учителей. Эту часть нужно разделить на 200 непересекающихся частей, так как каждый учитель должен учиться на своем наборе данных. Вторую часть считаем неразмеченной. Ее делим на два набора: для обучения модели студента и для тестирования работы. Важно понимать, что разметку обучающей выборки для модели студента выполняют модели учителей с использованием дифференциальной конфиденциальности.

transform = transforms.Compose([transforms.ToTensor(),

transforms.Normalize((0.5,), (0.5,)),

])

teacher\_data = datasets.MNIST('~/.pytorch/MNIST\_data/', download=True, train=True, transform=transform)

student\_data = datasets.MNIST('~/.pytorch/MNIST\_data/', download=True, train=False, transform=transform)

Начальная модель учителя и студента выглядит следующим образом:

self.fc1 = nn.Linear(784, 256)

self.fc2 = nn.Linear(256, 128)

self.fc3 = nn.Linear(128, 64)

self.fc4 = nn.Linear(64, 10)

**Ожидаемый результат**

1. Написан код, позволяющий подготовить исходный датасет для работы в рамках PATE
2. Написан код, позволяющий обучить модели учителей
3. Написан код, реализующий модель студента.
4. Итоговая метрика распознавания символов моделью студента должна быть не менее 80%.

Примерный код для решения лабораторной работы приведен в Jupyter Notebook, поставляемым вместе с данным руководством.

**Рекомендованная литература**

1. Dominik, Jens Sjlund, Tobias J. Oechtering. Decentralized Differentially Private Segmentation with PATE. <https://arxiv.org/pdf/2004.06567.pdf>
2. An Nguyen. Understanding Differential Privacy. <https://towardsdatascience.com/understanding-differential-privacy-85ce191e198a>
3. Материалы лекции по конфиденциальности данных при использовании ML.
4. Справочник по командам Python.