Лабораторная работа №2

Функции активации и оптимизаторы в PyTorch

Цель работы – изучить процесс построения и обучения простой нейронной сети в PyTorch на примере задачи классификации ботов, исследовать влияние различных функций активации и оптимизаторов на процесс обучения и качество модели.

Задание: выполнить формирование архитектуры нейронной сети для решения задачи классификации ботов, выполнить сравнительный анализ влияния различных функций активации и оптимизаторов на процесс обучения и качество модели:

1. в качестве исходных данных использовать датасет предыдущей ЛР с ботами и людьми, выполнить преобразование данных в тензор `torch.tensor`;
2. построить архитектуру нейронной сети через `nn.Linear`, как минимум с одним скрытым слоем;
3. в качестве функций активации попробовать по очереди: Sigmoid, Tanh, ReLU;
4. *Сравнительный анализ функции активации*: обучить сеть с каждой функцией активации (фиксируем оптимизатор, например SGD) и визуализировать loss и accuracy по эпохам;
5. *Сравнительный анализ оптимизаторов*: взять лучшую активацию по результатам предыдущего анализа, выполнить сравнение обучения при SGD, SGD (momentum=0.9), RMSprop, Adam, выполнить визуализацию loss/accuracy;
6. сделать вывод о наиболее подходящих функции активации и оптимизаторе для решения поставленной задачи;
7. составить отчет о проделанной работе в соответствии с требованиями кафедры.

Требования к отчету. Отчет должен содержать постановку задачи, исходные данные, результаты решения задачи, необходимые иллюстративные материалы.

Требования к защите

Защита лабораторной работы происходит индивидуально. Система оценки – рейтинговая.

Критерии оценки:

- корректность выполненного исследования;
- адекватность полученных результатов;
- качество отчета;
- качество ответов на контрольные вопросы;
- срок выполнения работы.

Время выполнения работы – 4 академических часа.

Контрольные вопросы

1. Зачем в нейронной сети нужны функции активации? Что произойдет, если использовать только линейные преобразования?
2. Почему при обучении модели необходимо обнулять градиенты перед каждым шагом оптимизации?
3. Как связаны между собой функция ошибки, градиентный спуск и оптимизатор?
4. Чем отличаются SGD и Adam с точки зрения обновления весов? Как влияет выбор оптимизатора на скорость сходимости и точность?
5. Почему иногда модель достигает высокой точности на обучающей выборке, но хуже работает на тестовой? Как это можно исправить?

Документация: https://docs.pytorch.org/docs/stable/torch.html