Матрицы Тёплица

Чернышов Игнат
Проект по курсу NLA

Вступление

Данный проект исследует связь между операциями свёртки в нейронных сетях и умножением на Тёплицевы матрицы. Основная часть проекта заключается в исследовании уже имеющихся результатов статьи «LU decomposition and Toeplitz decomposition of a neural network» и их локальной проверке.

Матрица тёплеца

Тёплицева матрица A называется матрицей A размера $n \times m$, которая удовлетворяет условию

$$\begin{pmatrix} t_0 & t_1 & t_2 & \cdots & t_{m-1} \\ t_{-1} & t_0 & t_1 & \cdots & t_{m-2} \\ t_{-2} & t_{-1} & t_0 & \cdots & t_{m-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{1-n} & t_{2-n} & t_{3-n} & \cdots & t_{m-n} \end{pmatrix}$$

- A имеет m столбцов и n строк
- $a_{i,j} = t_{j-i}$

Свойства матрицы Тёплица

- Свёртка сводится к матрице Тёплеца. $\kappa = (t_{m-1}, \ldots, t_0, t_{-1}, \ldots, t_{1-n})$, $\tilde{x} = (0 \ldots 0, x_1, \ldots, x_n, 0 \ldots 0)^T$

$$\kappa * \tilde{x} = \begin{pmatrix} t_{1-n} & \cdots & t_{-1} & t_0 & \cdots & t_{m-1} & 0 & \cdots & 0 \\ 0 & \cdots & t_{-2} & t_{-1} & \cdots & t_{m-2} & t_{m-1} & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & t_{1-n} & \cdots & t_{m-n} & \cdots & \cdots & t_{m-1} \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ x_1 \\ \vdots \\ x_n \\ 0 \\ \vdots \\ 0 \end{pmatrix} = Ax$$

- Информация о Тёплицевой матрице $A \in \mathbb{R}^{n \times m}$ хранится в векторе из $\mathbb{R}^{n+m-1}$
- Умножение матрицы Тёплецана вектор может осуществляться за $O(n \log n)$ операций

Реализация на языке Python

Используем библиотеку **torch**

```
class ToeplitzLinear(nn.Module):
    def __init__(self, in_features, out_features):
        super().__init__()
        self.diagonals = nn.Parameter(
            torch.empty(in_features + out_features - 1)
        )
        self.in_features = in_features
        self.out_features = out_features
        idx_matrix = torch.LongTensor([
            [i+j for j in range(in_features)]
            for i in range(out_features)
        ])
        self.reset_parameters()
        self.register_buffer("idx_matrix", idx_matrix)

    def reset_parameters(self):
        init.uniform_(self.diagonals, a=-1/10, b=1/10)

    def forward(self, x):
        W = self.diagonals[self.idx_matrix.to(x.device)]
        return x @ W.T
```

Пояснения

Код написан на языке Python с использованием библиотеки PyTorch. Код не является оптимальным, но показывает принцип работы. В частности код не оптимизирован для ускорения вычислений. Однако результаты показали, что даже такая реализация работает и достаточно хорошо.

За основу была взята структура **torch.nn.Linear**.

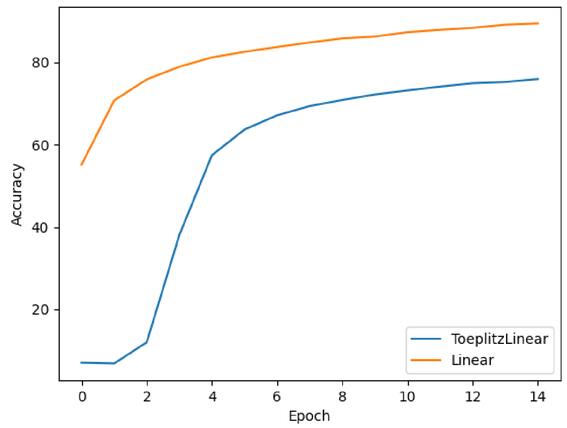
Вывод

Применив код к различным датасетам были получены следующие результаты:

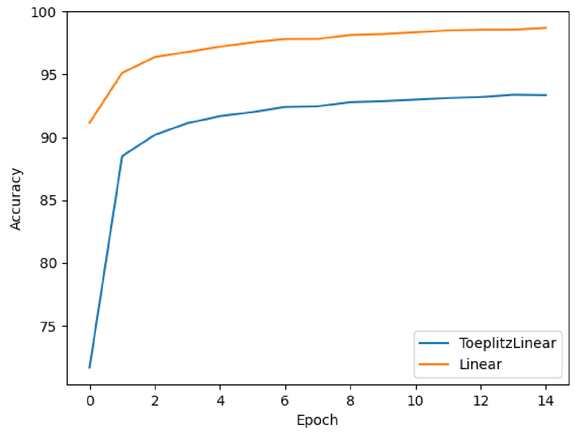
- На каждом датасете нейросеть обучалась (Проводилось 15 эпох обучения)
- Лучше всего показались результаты на датасетах с символами или очень простыми картинками (такие как **MNIST**, **SVHN** и **Fashion MNIST**)
- Хуже показались результаты на датасетах с более сложными картинками (такие как **CIFAR-10** и **STL-10**)

Результаты

Проверка работы на датасете **SVHN**



Проверка работы на датасете **MNIST**



Больше информации

Чтобы подробнее ознакомиться с кодом перейдите на мой гитхаб.