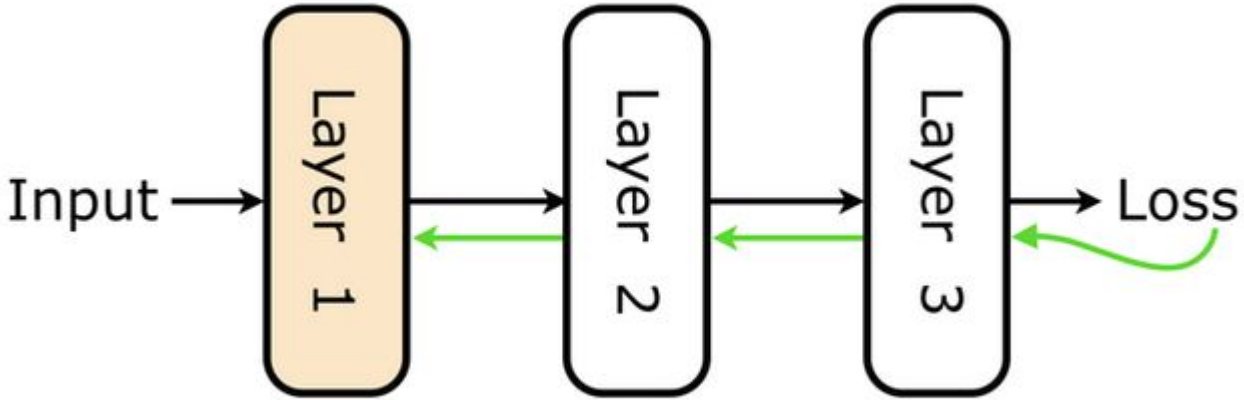
Синтетические градиенты

НИУ ВШЭ
Матковский Иван
14 февраля 2017

Обратное распространение ошибки



Вычислительные ограничения

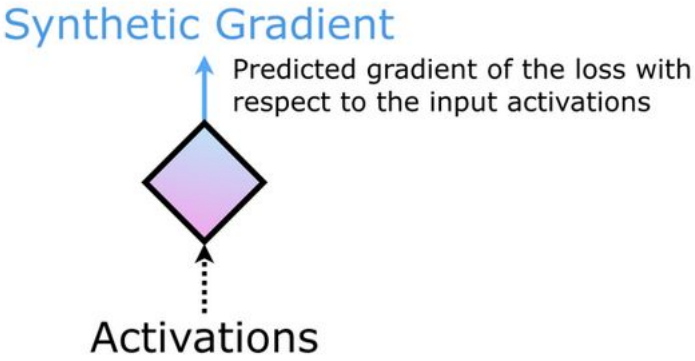
- **Forward Locking -** для вычисления выхода i - слоя, необходимо вычислить выходы предыдущих слоев
- **Update Locking -** для обновления коэффициентов весов на i - слое, необходимо вычислить выходы всех следующих слоев
- **Backwards Locking -** для обновления коэффициентов весов i - слоя, необходимо обновить следующие слоя

Посыл

- Все эти блокировки являются проблемами при конструировании распределенных систем для обработки нейронных сетей, так как они требуют синхронной работы

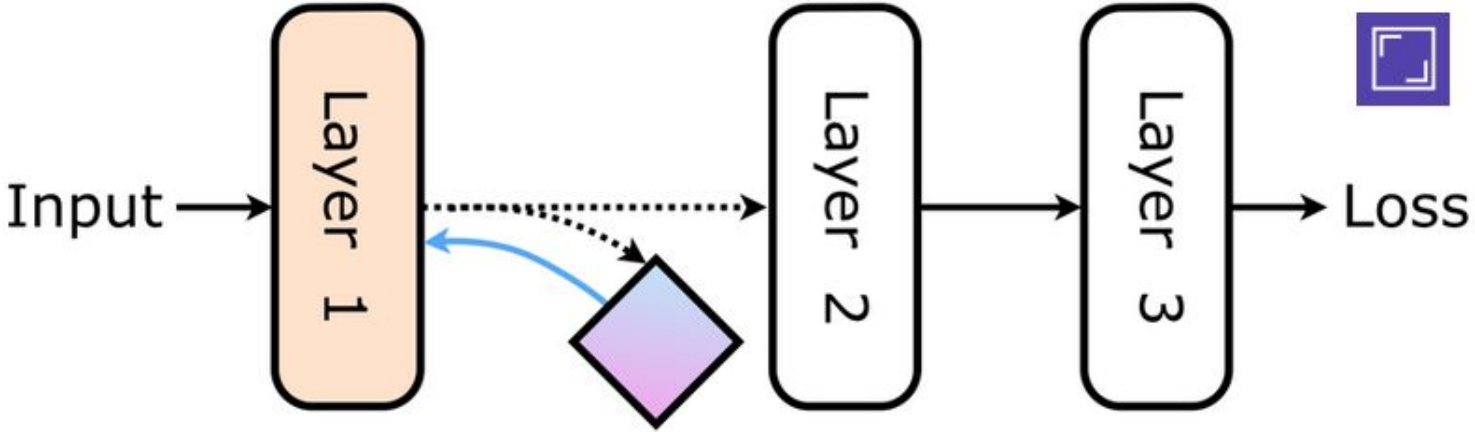
Синтетические градиенты

- Модель, позволяющая строить нейронные сети, в которых отсутствует Update Locking и Backwards Locking



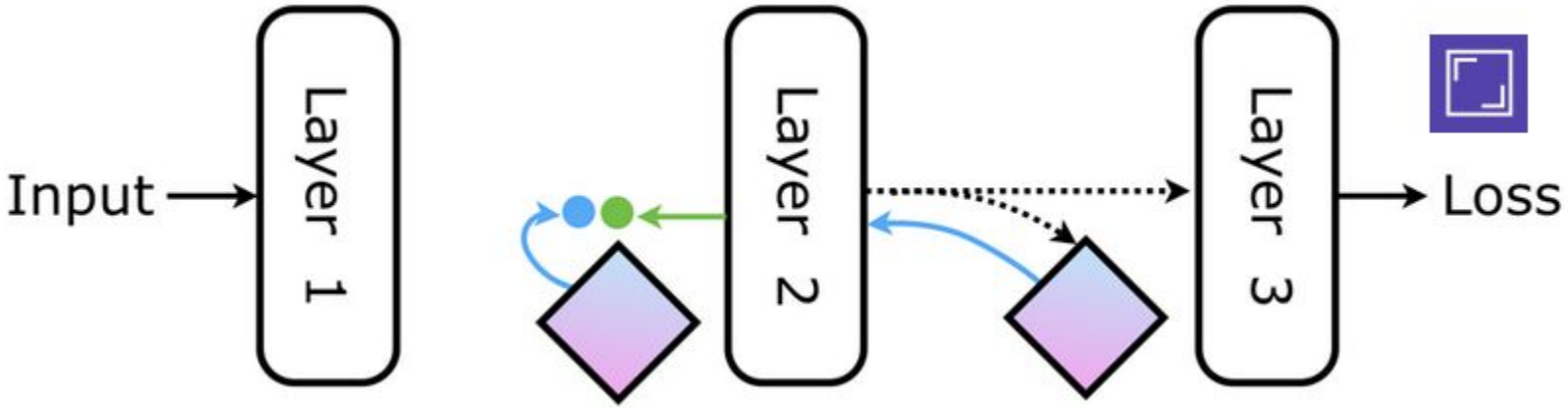
Описание модели

- Строится модель, которая учится предсказывать градиент по активациям слоя



Описание модели

- Обучение модели происходит следующим образом, к ней приходит либо градиент, распространенный со следующего слоя, полученный другой такой же моделью, либо реальные ошибки backpropagation



Сложность модели

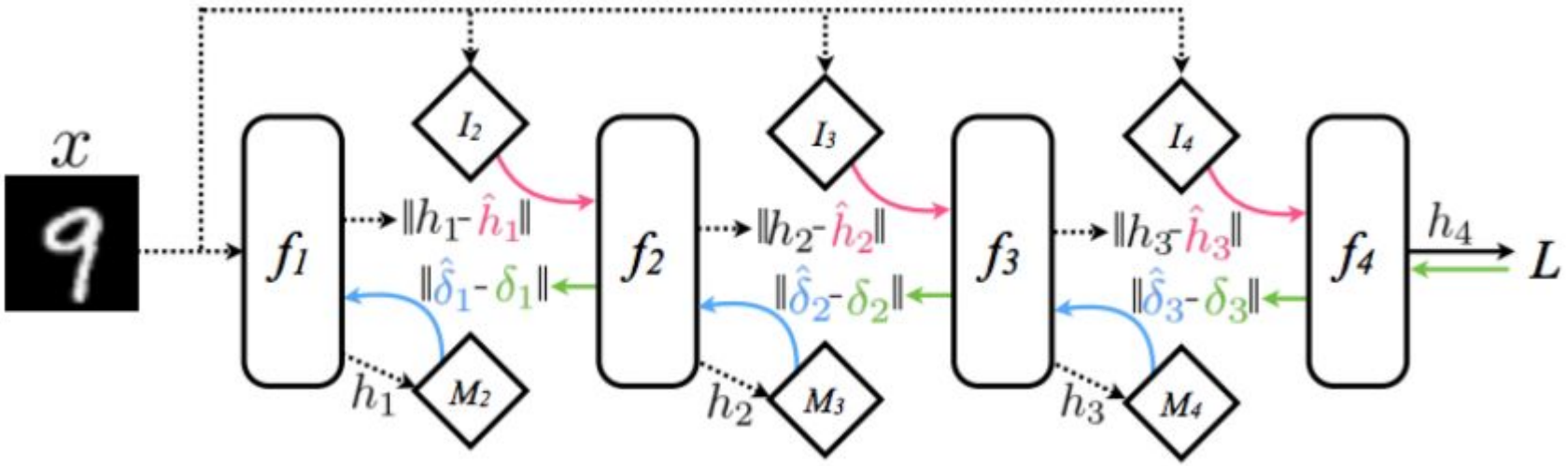
- Оказывается, что для построения такой модели требуется нейросеть с 0-3 скрытыми слоями. Даже линейная модель предсказывает градиент довольно хорошо, чтобы при использовании такой архитектуры можно было обучиться.

Качество модели

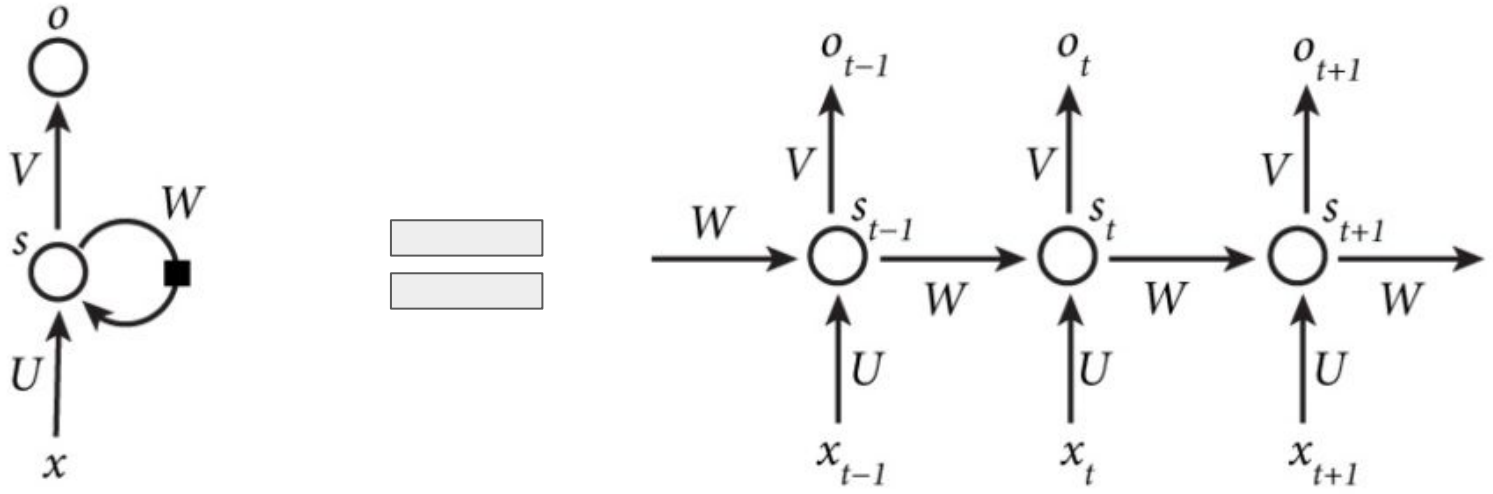
- Обучили сверточную нейронную сеть для CIFAR-10 классификации изображений, где каждый слой отделен с использованием синтетических градиентов. После 500k итераций получили такую же точностью, как и с использованием обратного распространения ошибки.

Complete Unlock

- Чтобы избавиться от Forward Locking можно использовать модель синтетического входа(аналогично синтетическим градиентам)



Архитектура рекуррентной сети

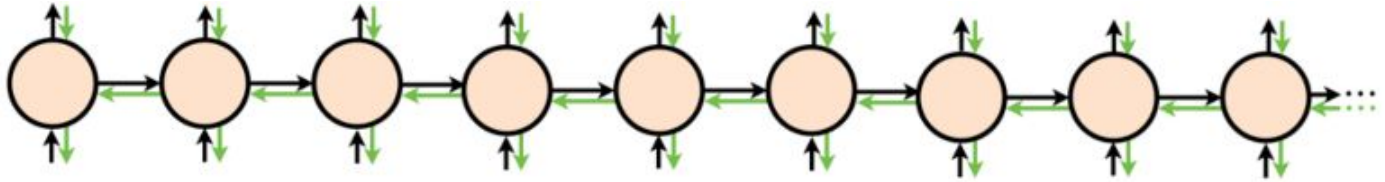


$$s_t = U x_t + W s_{t-1}$$

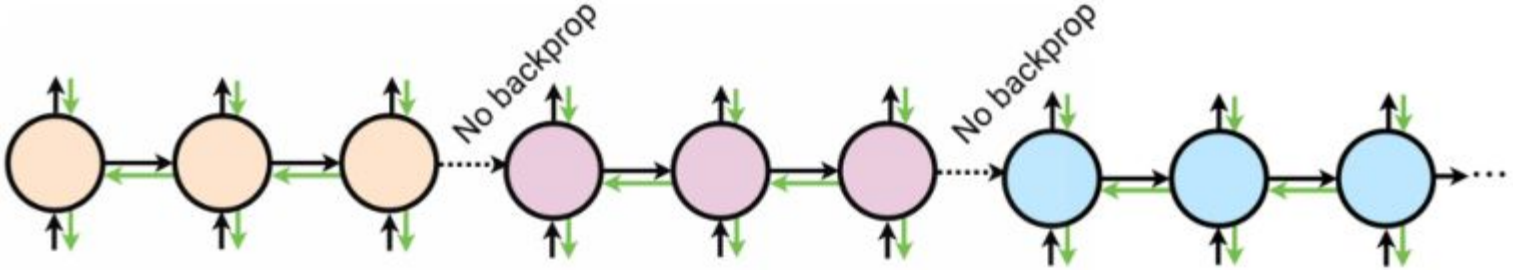
$$o_t = V s_t$$

Обучение рекуррентной сети

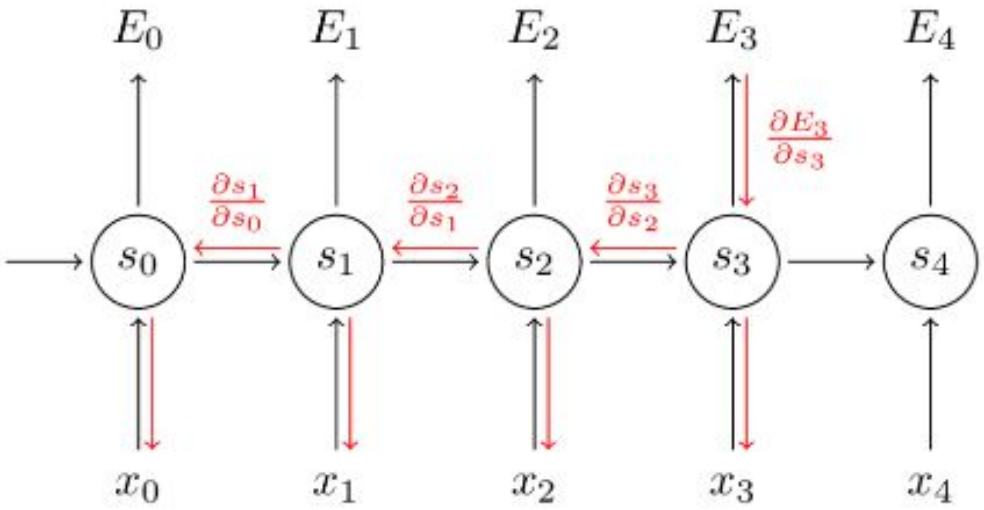
- Теоретически RNN, как выглядит, так и обучается обратным распространением ошибки по всей цепи(BBTT)



- На практике используется же лишь последнии i - шагов, это называется обучение методом усеченного обратного распространения ошибки (truncated BBTT)



Обучение рекуррентной сети



- Функция потерь: $E(y, o) = \sum_{i=k}^{k+3} \bar{E}(o_i, y_i)$, где $\bar{E}$ - функция потерь для одного элемента

Обучение рекуррентной сети

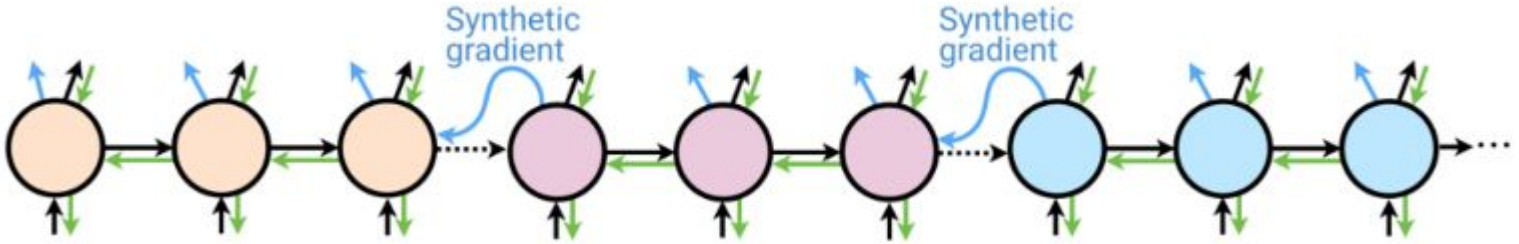
- Обучение каждого сегмента сети происходит независимо с помощью метода обратного распространения ошибки, так как функционал ошибки хорошо дифференцируется по W, V, U. Например дифференциал $\bar{E}(o_i, y_i)$ по W будет равен:

$$\frac{\partial E_{k+3}}{\partial W} = \sum_{i=k}^{k+3} \frac{\partial E_{k+3}}{\partial o_{k+3}} \frac{\partial o_{k+3}}{\partial s_{k+3}} \frac{\partial s_{k+3}}{\partial s_i} \frac{\partial s_i}{\partial W}$$

Для матриц U, V выписывается аналогично

Синтетический градиент в рекуррентной сети

- Идея заключается в том, чтобы предсказывать ошибку предыдущего сегмента сети с помощью синтетического градиента

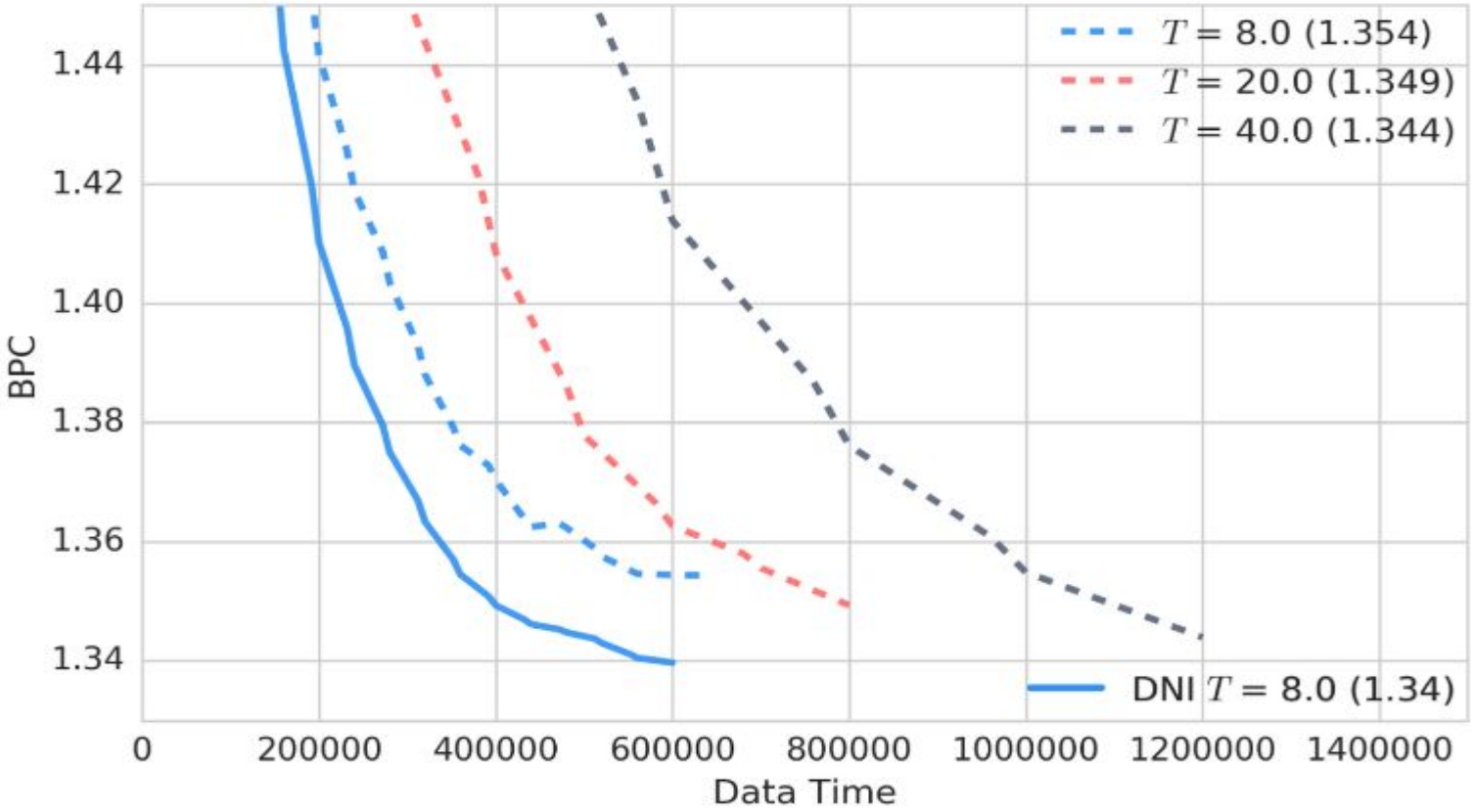


- Что позволяет не ограничиваться лишь текущим сегментом сети

Тестирование

- Датасет: Penn Treebank
- Задача: предсказать следующий символ

Тестирование



Спасибо за внимание