# Исследование применения boosting для сетей ResNet

Подготовил студент группы
ММО211С Дерявский Артем

# Постановка задачи

Цель работы: исследовать применение boosting для сетей ResNet.

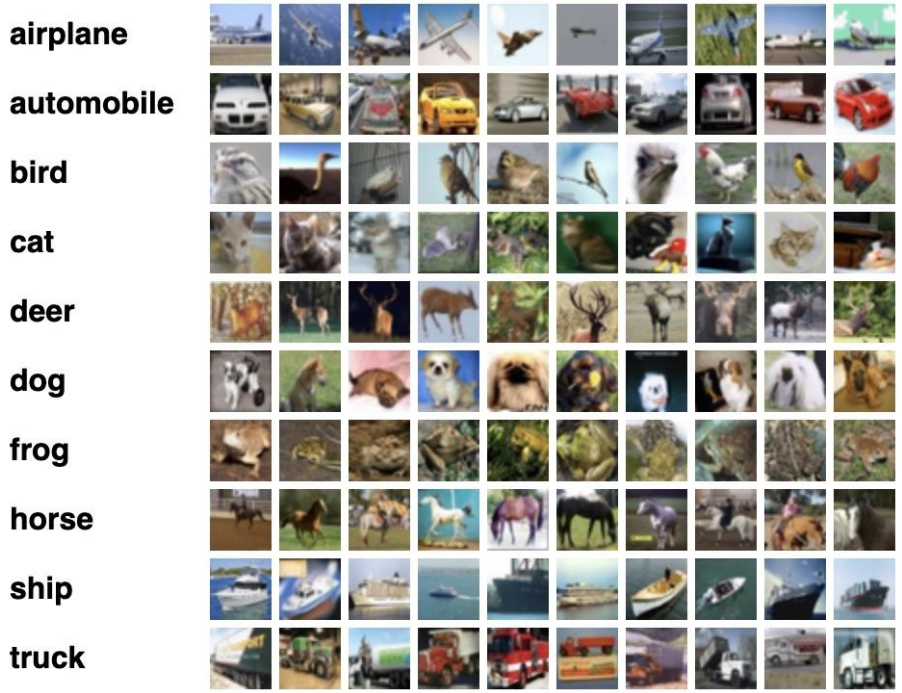
В основе работы лежит статья «Learning Deep ResNet Blocks Sequentially using Boosting Theory», в которой авторы предложили новый метод BoostResNet для обучения нейросетей с остаточными блоками, который обобщается на более широкий класс алгоритмов и является альтернативой end-to-end back propagation.

В общем случае алгоритм строит модель, последовательно обучая составляющие ее блоки, задавая глубину, опираясь на заданное условие останова. При этом модель может достигнуть слишком большой глубины. Поэтому в рамках данной работы будет сымитирована ситуация, при которой мы имеем ограниченные ресурсы (бюджет) и применим BoostResNet к ResNet50.

Задачи работы:

1. Реализовать алгоритм BoostResNet и применить его для обучения ResNet50
2. Исследовать данный алгоритм на предмет скорости обучения, требований к оперативной памяти и точности предсказания класса на датасетах CIFAR-10 и SVHN.

# Датасеты




CIFAR-10
50 000 картинок в тренировочном наборе
10 000 картинок в тестовом наборе.
Размер изображений 32x32
Во время обучения к изображениям применялась аугментация.

SVHN
73257 картинок в тренировочном наборе
26032картинок в тестовом наборе.
Размер изображений 32x32

# Описание бейзлайна

В качестве бейзлайна была взята сеть ResNet50, состоящая из 17 блоков (включая первую свертку) с классическим методом обучения end-2-end back propagation.

Размер батча: 512

Оптимизатор: (CIFAR-10) SGD с моментом 0.9 и L2-регуляризацией. Начальный learning rate 0.1

(SVHN) Adam с L2-регуляризацией, learning rate 0.001

Функция потерь: кросс-энтропия

Корректировка шага обучения: косинусный отжиг (только для CIFAR-10)

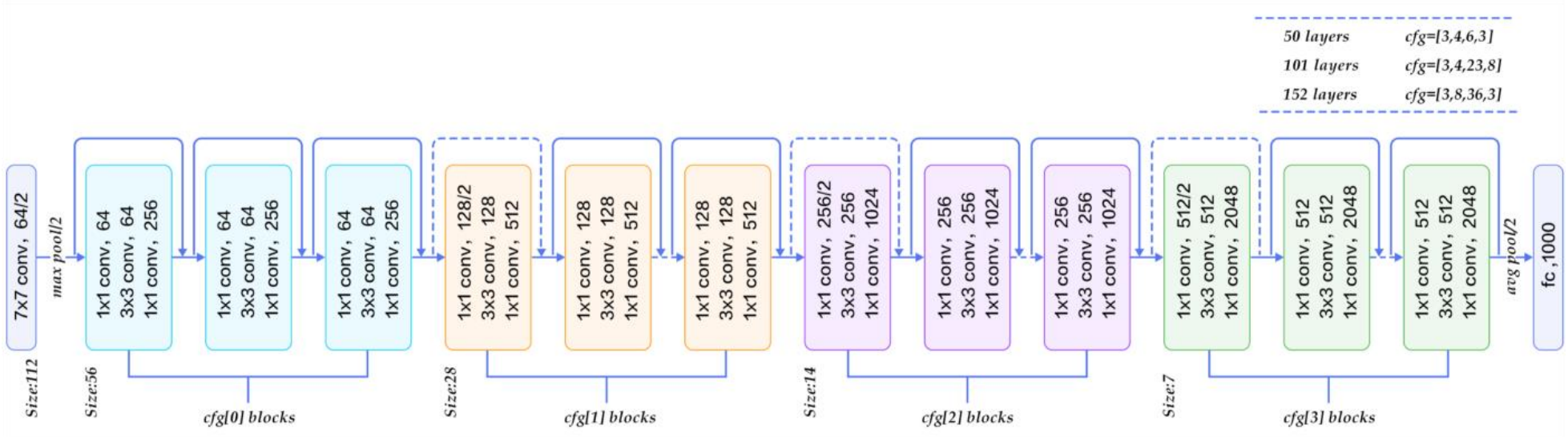



Рис.1 Архитектура ResNet без поправки на количество блоков во второй и третьей группе.

# Процесс обучения ResNet50 (e2eBP)

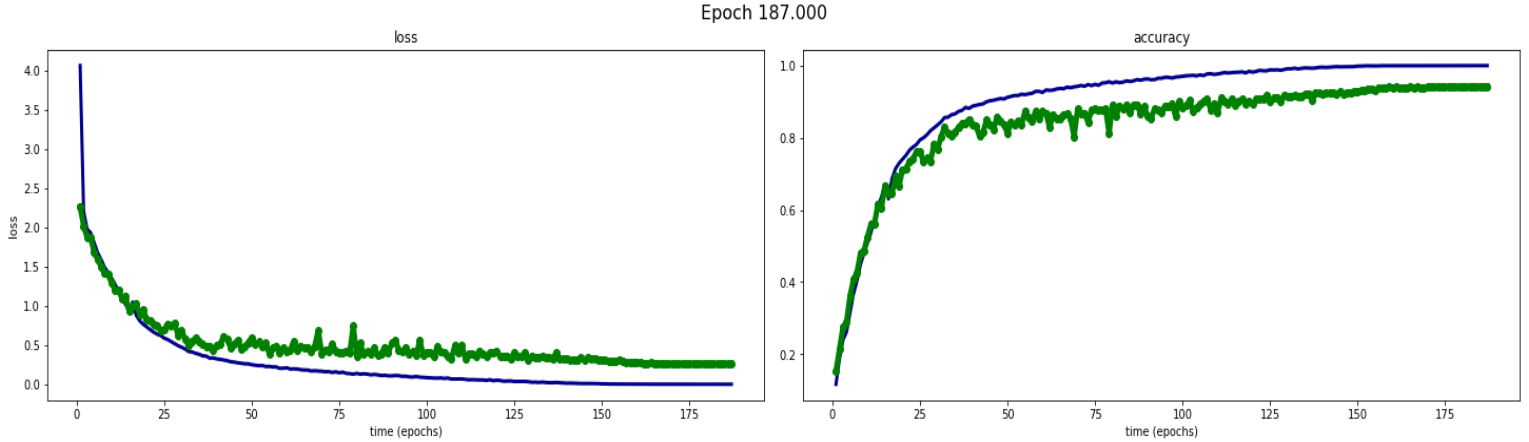



Рис 2. Результаты обучения ResNet50 + e2eBP на датасете CIFAR-10

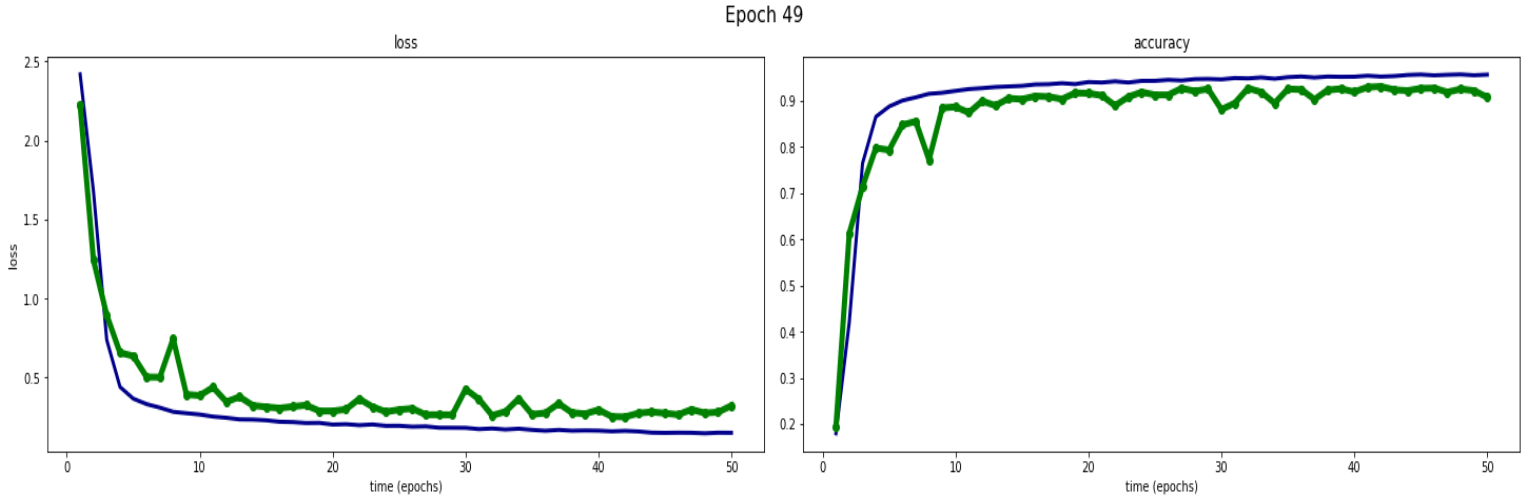
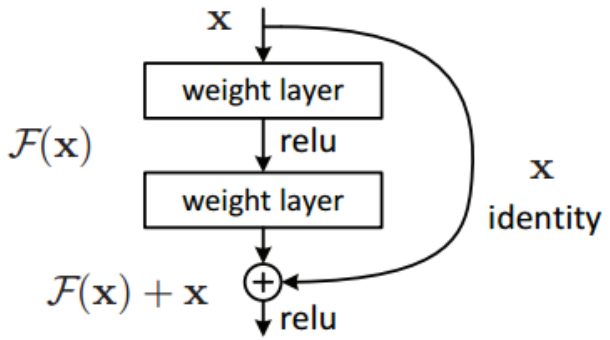



Рис 3. Результаты обучения ResNet50 + e2eBP на датасете SVHN

Легенда: слева график функций потерь (зеленый – валидационное множество, синее - тренировочное), справа аналогично график значения точности.

# BoostResNet: метод

- Пытаемся применить теорию бустинга к сетям ResNet
- Сложность в том, что необходимо буститься по признакам
- BoostResNet последовательно обучает T-неглубоких ResNet сетей
- Как известно, ResNet состоит из остаточных блоков, а каждый блок состоит из блока нейросети и identity loop:




- Для каждого блока t введем вспомогательный классификатор:

$$o_t(x) \stackrel{\text{def}}{=} \mathbf{w}_t^\top g_t(x) \in \mathbb{R}$$

# BoostResNet: расхождения с оригиналом

Посмотрим на псевдокод алгоритма, предлагаемого в статье

**Algorithm 3** BoostResNet: telescoping sum boosting for multi-class classification

**Input:** Given $(x_1, y_1), \dots (x_m, y_m)$ where $y_i \in \mathcal{Y} = \{1, \dots, C\}$ and a threshold $\gamma$

**Output:** $\{f_t(\cdot), \forall t\}$ and $W_{T+1}$ ▷ Discard $w_{t+1}, \forall t \neq T$

Initialize $t \leftarrow 0, \tilde{\gamma}_0 \leftarrow 1, \alpha_0 \leftarrow 0, o_0 \leftarrow \mathbf{0} \in \mathbb{R}^C, s_0(x_i, l) = 0, \forall i \in [m], l \in \mathcal{Y}$

Initialize cost function $\mathrm{C}_0(i,l) \leftarrow \begin{cases} 1 & \text{if } l \neq y_i \\ 1 - C & \text{if } l = y_i \end{cases}$

**while** $\gamma_t > \gamma$ **do**

  $f_t(\cdot), \alpha_{t+1}, W_{t+1}, o_{t+1}(x) \leftarrow$ Algorithm 4$(g_t(x), \mathrm{C}_t, o_t(x), \alpha_t)$

  Compute $\gamma_t \leftarrow \sqrt{\frac{\tilde{\gamma}_{t+1}^2 - \tilde{\gamma}_t^2}{1 - \tilde{\gamma}_t^2}}$ ▷ where $\tilde{\gamma}_{t+1} \leftarrow \frac{-\sum_{i=1}^{m} \mathrm{C}_t(i,:) \cdot o_{t+1}(x_i)}{\sum_{i=1}^{m} \sum_{l \neq y_i} \mathrm{C}_t(i,l)}$

  Update $s_{t+1}(x_i, l) \leftarrow s_t(x_i, l) + h_t(x_i, l)$ ▷ where $h_t(x_i, l) = \alpha_{t+1} o_{t+1}(x_i, l) - \alpha_t o_t(x_i, l)$

  Update cost function $\mathrm{C}_{t+1}(i,l) \leftarrow \begin{cases} e^{s_{t+1}(x_i,l) - s_{t+1}(x_i,y_i)} & \text{if } l \neq y_i \\ -\sum_{l' \neq y_i} e^{s_{t+1}(x_i,l') - s_{t+1}(x_i,y_i)} & \text{if } l = y_i \end{cases}$

  $t \leftarrow t + 1$

**end while**

$T \leftarrow t - 1$

**Algorithm 4** BoostResNet: oracle implementation for training a ResNet module (multi-class)

**Input:** $g_t(x), s_t, o_t(x)$ and $\alpha_t$

**Output:** $f_t(\cdot)$, $\alpha_{t+1}$, $W_{t+1}$ and $o_{t+1}(x)$

$(f_t, \alpha_{t+1}, W_{t+1}) \leftarrow \arg\min_{(f,\alpha,V)} \sum_{i=1}^{m} \sum_{l \neq y_i} e^{\alpha V^\top [f(g_t(x_i), l) - f(g_t(x_i), y_i) + g_t(x_i, l) - g_t(x_i, y_i)]}$

$o_{t+1}(x) \leftarrow W_{t+1}^\top [f_t(g_t(x)) + g_t(x)]$

В оригинале процесс обучения был завязан на такой величине как $\gamma_t$, это слабое условие обучение слабого модульного классификатора $h_t$. Оно зависит от сильных условий обучения $\tilde{\gamma}_t$, которые характеризуют вспомогательные классификаторы. Но в условиях ограниченных ресурсов и заданной архитектуры нет смысла и возможности ориентироваться на данную метрику. Поэтому было принято решение упростить данную часть метода.

# Реализация BoostResNet

- Для экспериментов была реализована упрощенная имплементация алгоритма BoostResNet, в которой мы заранее задаем размер модели. Как увидим далее на ее производительность данное упрощение не повлияет.
- Берем Resnet-50 и делим ее на 17 блоков согласно ее стандартной архитектуре. Обучаем блоки последовательно, добавляя к каждому текущему блоку вспомогательный классификатор
- Обучаем блоки как обычно
- Отдельно стоит вопрос о том, когда останавливать обучение блока, но правильный рецепт так и не был найден

# Процесс обучения BoostResNet

В ходе экспериментов выяснилось, что отдельные блоки быстро переобучаются и для их обучения достаточно одной эпохи. Время обучения каждого блока составило от 7 до 70 секунд, что значительно быстрее, чем при e2e-back propagation, где каждая эпоха занимала 82 секунды.
Оптимизатор: AdamW с настройками по умолчанию

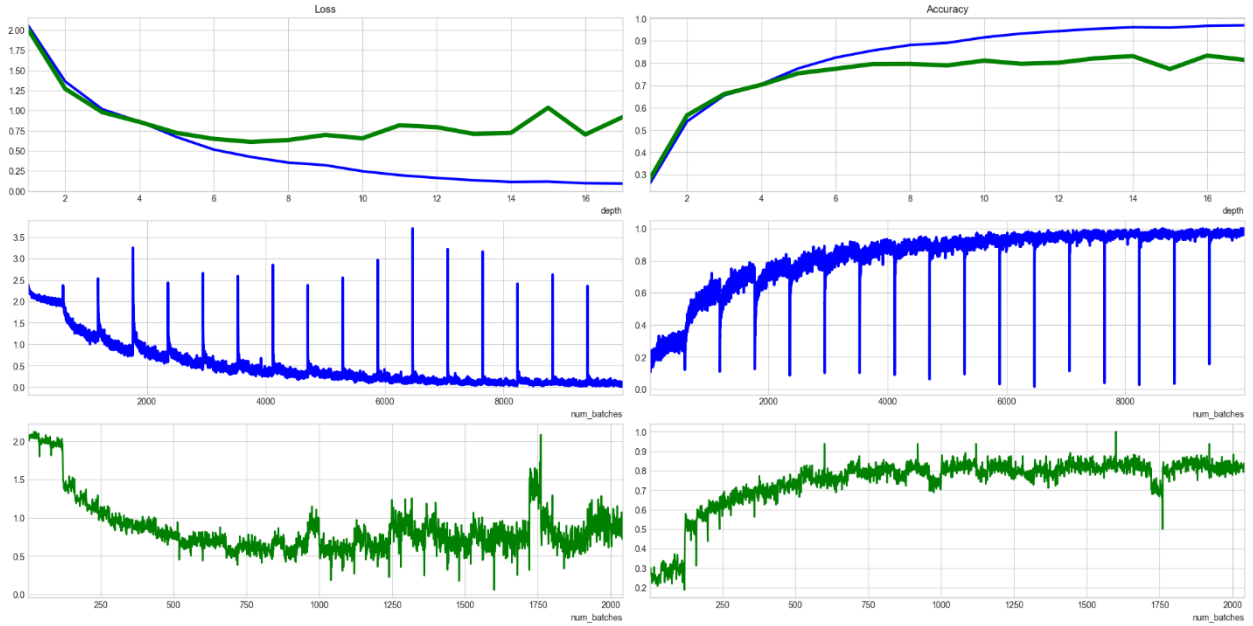



Рис. Процесс обучения на CIFAR-10

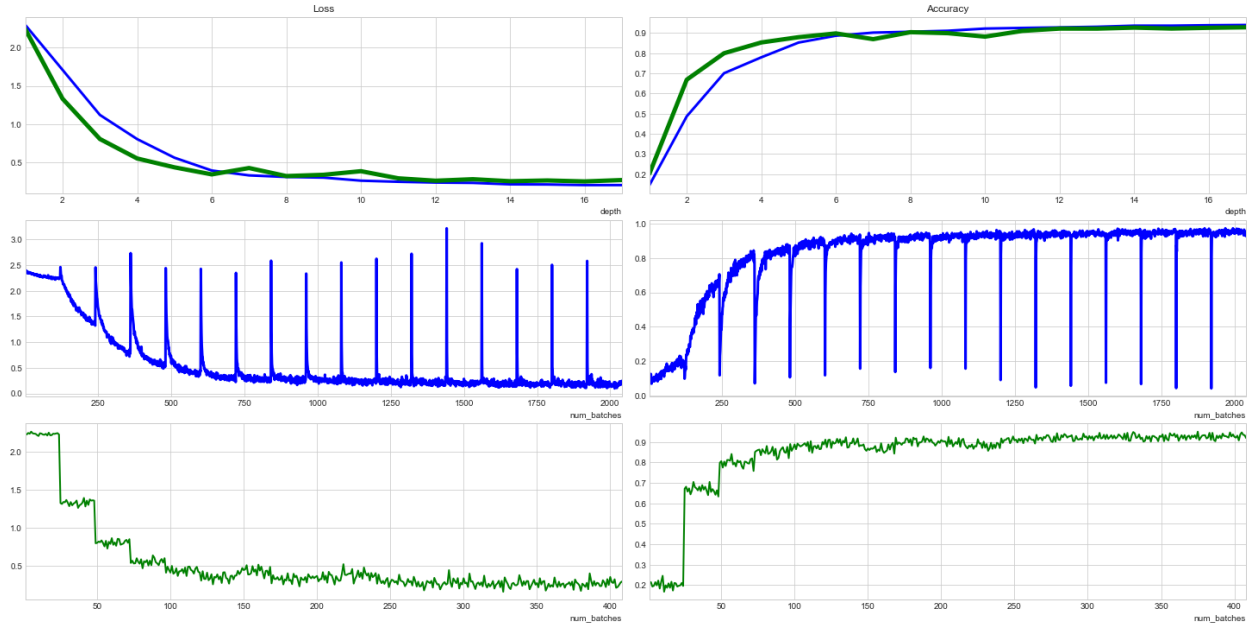



Рис. Процесс обучения на SVHN

Описание графиков: синий – train, зеленый – test. Верхний горизонтальный ряд Loss и Accuracy в зависимости от глубины. Второй и третий ряды те же метрики в зависимости от номера батча, чтобы посмотреть как ведут себя метрики на всем протяжении процесса обучения.

# BoostResNet + e2eBP

Возможно, BoostResNet стоит использовать как метод преобучения для общего ускорения обучения ResNet-подобных архитектур.

Подобная методика обучения применялась только для датасета CIFAR-10.

Оптимизатор AdamW с настройками по умолчанию.

Корректировка шага: косинусный отжиг.

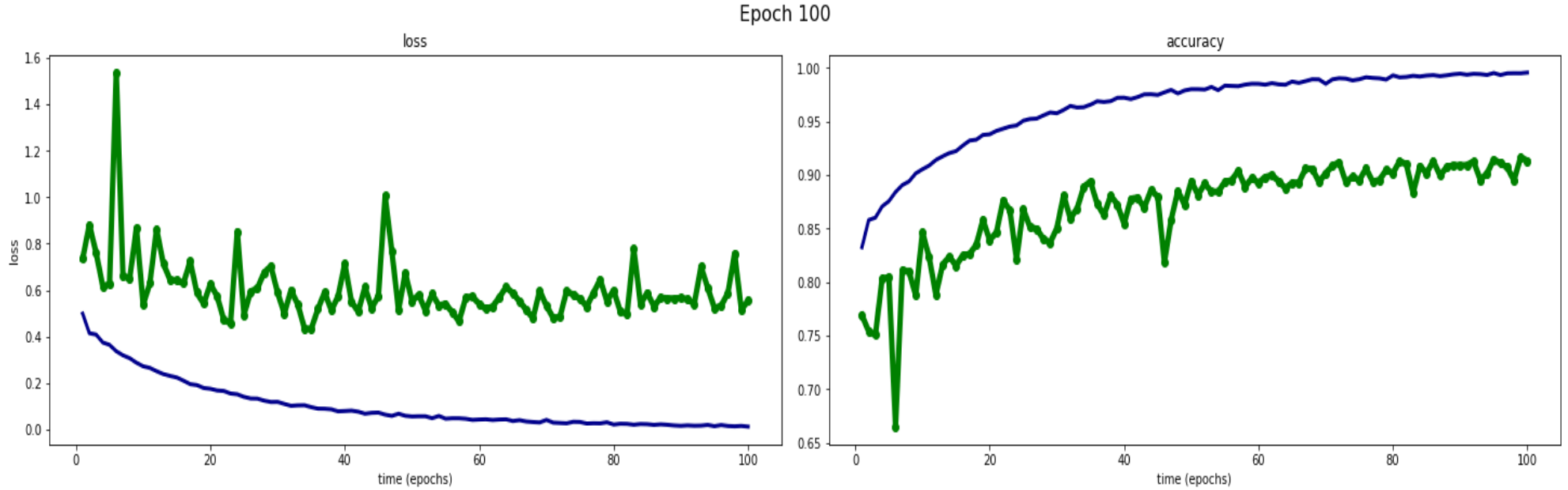



Рис. Процесс дообучения ResNet50 на датасете CIFAR-10

# Итоговые результаты

| | | e2eBP | BoostResnet | BoostResNet + e2eBP | BoostResNet (статья) | BoostResNet + e2eBP (статья) |
|---|---|---|---|---|---|---|
| CIFAR-10 | Train acc. | 1 | **0,96** | **0,99** | 0,921 | 0,996 |
| | Test acc. | 0,94 | **0,83** | **0,916** | 0,821 | 0,881 |
| | Потребление памяти, Гбайт | 11,3 | 4,5*<br>13,6** | 4,5*<br>13,6** | - | - |
| SVHN | Train acc. | 0,95 | 0,94 | - | 0,969 | - |
| | Test acc. | 0,93 | **0,93** | - | 0,938 | - |
| | Потребление памяти, Гбайт | 11,3 | 4,5*<br>13,6** | 4,5*<br>13,6** | - | - |

Табл 1. Итоговые результаты экспериментов.

*- в VRAM находился только текущий обучаемый блок и классификатор

**- в VRAM находилась вся сеть ResNet50

# Выводы

По результатам проведенного исследования можно сделать следующие выводе о применимости BoostResNet в условиях ограниченных ресурсов:

1) BoostResNet можно применить при сильном дефиците видеопамяти. В VRAM может находится только текущий обучаемый блок. (Не рекомендуется, так как много работы ляжет на CPU)

2) **Ускорение обучения**. BoostResNet можно применить для первичной тренировки модели, которую затем можно дообучить, используя классический back propagation, при этом сократив число эпох.

# Доп. Материалы

- Репозиторий проекта на GitHub