МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Механико-математический факультет

Кафедра математической теории интеллектуальных систем

Курсовая работа

ПРУНИНГ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ЗАДАЧ КОМПЬЮТЕРНОГО ЗРЕНИЯ

NEURAL NETWORK PRUNING FOR COMPUTER VISION TASKS

Подготовил:

Студент 305 группы

Харитонов Денис Александрович

Научный руководитель:

Половников Владимир Сергеевич

Москва, 2024г.

1. Введение

В последние годы нейронные сети стали основным подходом для задач в области компьютерного зрения, таких как классификация изображений, детекция объектов и семантическая сегментация. Одной из ключевых проблем, с которыми сталкиваются исследователи и разработчики в этой области, является необходимость значительных вычислительных ресурсов для обучения и развертывания глубоких моделей. Эти ресурсы включают не только мощное аппаратное обеспечение, но и большое количество памяти и энергии, что делает использование таких моделей неэффективным и затратным, особенно в мобильных и встроенных системах.

Тема оптимизации нейронных сетей становится все более актуальной по нескольким причинам:

1. **Эффективность и скорость**: Уменьшение количества параметров сети позволяет ускорить процесс инференса, что критически важно для приложений реального времени, таких как автономные транспортные средства и системы видеонаблюдения.
2. **Снижение энергопотребления**: Меньшее количество вычислений и памяти снижает энергопотребление, что является важным фактором для мобильных устройств и устройств IoT с ограниченным источником питания.
3. **Доступность и экономичность**: Сжатие моделей делает возможным использование сложных нейронных сетей на более дешевых и менее мощных устройствах, расширяя область их применения и делая передовые технологии доступными для более широкого круга пользователей.
4. **Поддержка устойчивого развития**: Снижение потребления ресурсов способствует более экологически устойчивому использованию вычислительных технологий, что соответствует современным трендам в области "зеленых" технологий и уменьшения углеродного следа.

Для решения этих проблем можно использовать разные подходы, такие как квантизация [1], дистилляция [2] и прунинг [3]. Квантизация предполагает уменьшение разрядности чисел, используемых для представления параметров модели, что значительно снижает требования к памяти и вычислительным ресурсам. Дистилляция включает обучение небольшой модели с использованием знаний, извлеченных из более крупной и сложной модели, что позволяет создать более компактные и быстрые модели с сохранением высокой точности. Прунинг же представляет собой удаление несущественных параметров или целых слоев (карт) в нейронной сети.

В рамках этой работы нами был проведен сравнительный анализ различных техник неструктурированного прунинга на примере сверточной нейронной сети для классификации изображений ResNet-34 [4] на датасетах CIFAR10 и CIFAR100 [5].

2. Описание основных техник прунинга

Прунинг можно разделить на структурированный и неструктурированный. [3]

- Структурированный прунинг предполагает удаление целых строк или столбцов весов, что уменьшает размерность матриц весов. Это приводит к удалению нейронов вместе со всеми их входящими и исходящими связями в полносвязных слоях или целых сверточных фильтров в сверточных слоях.
- Неструктурированный прунинг заключается в обнулении отдельных весов без изменения размерности матриц весов. Это приводит к обнулению отдельных связей между нейронами в полносвязных слоях или отдельных весов в сверточных фильтрах.

Хотя неструктурированный прунинг позволяет добиться большего количества нулевых весов, его реализация дает меньше преимуществ. Это связано с тем, что современные вычисления обычно параллелизованы и выполняются на уровне векторов и матриц. Специальные структуры данных для разреженных матриц могут дать преимущество только при очень высокой степени разреженности (менее 10% ненулевых элементов).

В зависимости от подхода к процессу удаления весов прунинг можно разделить на следующие категории [3]:

- Однократный прунинг предполагает, что предварительно обученная модель в определенный момент подвергается разовому удалению части весов согласно выбранному критерию. После этого, чтобы компенсировать возможное падение качества, модель дообучают в течение некоторого количества эпох, чтобы приблизить её производительность к исходному уровню.
- Итеративный прунинг осуществляется поэтапно. На каждом этапе из модели удаляется часть весов, после чего проводится дообучение. Этот процесс повторяется многократно, пока модель не достигнет желаемого уровня разреженности. Данный подход сложнее в реализации и требует больше времени, однако позволяет более плавно адаптировать модель к потере весов и в конечном счете получить лучший результат.
- Кроме того, можно начинать обучение сразу с разреженной модели и постепенно удалять наименее важные веса, одновременно добавляя новые, которые ранее были обнулены, на основе определенного критерия. Преимущество этого подхода заключается в возможности обучения моделей, которые в плотном виде (dense) не помещаются в память устройства.

- Также существуют методы, основанные на генетических алгоритмах [6]. Популяция создается из нескольких прореженных версий нейронной сети, и каждая обучается отдельно. Новые сети создаются с использованием мутаций, скрещивания и селекции. Затем популяции вознаграждаются за меньшее количество параметров и за лучшее обобщение. Однако, этот подход непрактичен для современного современных глубоких моделей из-за высокой сложности обучения ансамблей моделей.

Одним из самых простых и эффективных подходов к выбору весов для удаления является прунинг на основе абсолютной величины (magnitude-based pruning) [7], который удаляет веса или карты с наименьшей L1 нормой. Несмотря на свою простоту, этот метод показывает хорошие результаты на практике. Благодаря легкости реализации и невысоким затратам на использование, этот способ прореживания является самым популярным на данный момент. Дополнительные расходы при обучении связаны только с хранением масок для неудаленных весов. Существуют также адаптации этого алгоритма, использующие другие нормы (например, L2), но важно отметить, что этот метод предполагает, что значения входов весов находятся примерно в одном диапазоне.

Кроме того, нейронную сеть можно прореживать в зависимости от чувствительности выходов к значениям конкретных весов:

- На основе дисперсии выходов нейрона [8]: Если выход нейрона мало изменяется для различных входных данных, этот нейрон можно удалить, добавив его выход в bias для следующих нейронов. Например, если выход нейрона приблизительно равен константе C, можно заменить этот нейрон на константу.
- На основе взаимной корреляции между весами [9]: Если в слое есть сильно коррелированные нейроны, соединенные с одними и теми же нейронами в следующем слое, их можно объединить в один нейрон, что уменьшит количество операций.
- Использование разложения Тейлора для функции потерь [10]: В этом подходе функция потерь раскладывается в ряд до первого или второго члена, и выбираются веса, обнуление которых минимально влияет на функцию потерь. Недостатком данного метода является высокая вычислительная сложность при расчете матрицы Гессе, а также то, что разложение Тейлора точно только в малой окрестности точки.

3. Описание модели и датасетов

ResNet-34 (Residual Network с 34 слоями) — это глубокая сверточная нейронная сеть для классификации изображений. Основной инновацией сетей семейства ResNet является использование остаточных блоков (residual blocks), которые помогают сети легче обучаться за счет соединений быстрого доступа (skip connections).

Архитектура ResNet-34:

У нейронной сети ResNet-34 63.5 миллиона параметров.

Входной слой: Принимает изображения размером 224x224 и выполняет начальную свертку с ядром 7x7 и шагом 2, после чего следует операция субдискретизации (subsampling) с помощью слоя пулинга с функцией максимума 3x3.

Остаточные блоки: Сеть состоит из нескольких групп остаточных блоков, каждая из которых содержит по 2 или 3 сверточных слоя. Эти блоки имеют прямые связи, которые позволяют градиенту эффективно распространяться через сеть во время обучения. В ResNet-34 используется следующая конфигурация блоков:

- 3 блока с 64 фильтрами
- 4 блока с 128 фильтрами
- 6 блоков с 256 фильтрами
- 3 блока с 512 фильтрами

Промежуточные слои: Между группами остаточных блоков расположены слои пулинга (stride 2), которые уменьшают размер пространственных измерений изображения.

Выходной слой: После всех остаточных блоков применяются пулинг с функцией среднего (global average pooling) и полносвязный (fully connected) слой с 1000 нейронами (в исходной версии для ImageNet). В случае использования ResNet-34 для других датасетов, этот полносвязный слой можно адаптировать под нужное количество классов.

Датасет CIFAR-10 состоит из 60 000 цветных изображений размером 32x32 в 10 классах, по 6000 изображений в каждом классе. Он разделен на 50 000 изображений в тренировочной выборке и 10 000 в тестовой. В обучающей и тестовой выборке по 5000 и 1000 изображений каждого из десяти классов.

Датасет CIFAR100 аналогичен CIFAR-10, за исключением того, что он содержит 100 классов, каждый из которых содержит 600 изображений. В

каждом классе имеется 500 обучающих изображений и 100 тестовых изображений. 100 классов CIFAR-100 сгруппированы в 20 суперклассов.

4. Описание экспериментов

Для оценки эффективности различных методов прунинга нами была проведена серия экспериментов с использованием модели ResNet-34 на датасетах CIFAR10 и CIFAR100. Эти эксперименты были выполнены с использованием фреймворка PyTorch, который предоставляет удобные инструменты для реализации и обучения глубоких нейронных сетей.

Целью наших экспериментов было исследование влияния различных техник прунинга на точность модели и её производительность. Мы рассмотрели однократный и итеративный прунинг, а также технику обучения изначально разреженной модели.

4.1 Подготовка данных

Аугментации для CIFAR10 и CIFAR100 были одинаковыми.

На тренировочной выборке использовались следующие аугментации:

1. Сначала при помощи билинейной интерполяции [11] размер изображений увеличивался до 224x224 (класс torchvision.transforms.Resize)
2. Далее использовался класс torchvision.transforms.AutoAugment [12], автоматически подбирающий лучшие аугментации. Для датасетов CIFAR10 и CIFAR100 это преимущественно цветовые искажения, такие как изменение цветового баланса, контрастности и яркости.
3. Затем проводилась нормализация (torchvision.transforms.Normalize) по каждому каналу путем вычитания среднего значения и деления на стандартное отклонение тренировочной выборки.

На валидационной выборке проводились только изменение размера до 224x224 и нормализация со средним и стандартным отклонением, вычисленными на тренировочной выборке.

4.2 Обучение базовых моделей

Схема обучения базовой модели для CIFAR10 и CIFAR100 одинаковая, за исключением количества эпох: 20 для CIFAR10, и 30 для CIFAR100. Мы использовали предварительно обученную на датасете ImageNet [13] модель ResNet-34 и заменили её выходной линейный слой на слой с количеством нейронов, соответствующим числу классов (10 для CIFAR10 и 100 для CIFAR100) и инициализировав его из равномерного распределения Ксавье [14]. Дообучение производилось с помощью оптимизатора Adam [15] с параметрами learning rate 1e-5 для сверточных слоев и 1e-4 для линейного слоя, $\beta_1 = 0.9$, $\beta_2 = 0.999$ и weight decay 5e-4. Более высокий learning rate для

полносвязного слоя обусловлен тем, что сверточные слои уже обучены извлекать признаки, а полносвязный слой нужно обучать с нуля.

С помощью данного способа обучения модель ResNet-34 достигла точности 96% на CIFAR10 и 84% на CIFAR100, что является высоким показателем среди сетей данного семейства.

4.3 Прунинг

В ходе исследования нами были проведены следующие эксперименты. Во всех них веса удалялись только из сверточных слоев.

На датасете CIFAR10:

4.3.1 Однократный прунинг 99.5% весов с последующим дообучением

- Конфигурация:
  - Точность модели до прунинга 96%
  - Критерий отбора карт для удаления: L1 норма
  - Loss функция: кросс-энтропия
  - Оптимизатор: Adam
  - Learning rate: 3e-4, снижаемый в 10 раз на 20-й эпохе с помощью torch.optim.lr_scheduler.MultiStepLR
  - Weight decay: 5e-4
  - Длительность: 40 эпох
- Результаты:
  - Изначальная точность сразу после удаления весов оказалась на уровне случайного угадывания
  - Итоговая точность после файн-тюнинга составила 89%, что говорит о том, что модель сохраняет большую часть своей функциональности даже при высокой степени разреженности.

4.3.2 Однократный прунинг 95% весов с последующим дообучением

- Конфигурация:
  - Точность модели до прунинга 96%
  - Критерий отбора карт для удаления: L1 норма
  - Loss функция: кросс-энтропия
  - Оптимизатор: Adam
  - Learning rate: 3e-4, снижаемый в 10 раз на 30-й эпохе с помощью torch.optim.lr_scheduler.MultiStepLR
  - Weight decay: 5e-4
  - Длительность: 40 эпох
- Результаты:

- Изначальная точность сразу после удаления весов оказалась на уровне случайного угадывания
- Итоговая точность после файн-тюнинга составила 96%, что демонстрирует эффективность метода для умеренного уровня прунинга.

4.3.3 Шесть итераций с удалением равной доли весов и дообучением на каждой начиная с уровня разреженности 0.95 до достижения уровня разреженности 0.995.

- Конфигурация:
  - Точность модели до прунинга 96%
  - Критерий отбора карт для удаления: L1 норма
  - Loss функция: кросс-энтропия
  - Оптимизатор: Adam
  - Learning rate: 1e-4
  - Weight decay: 5e-4
  - Длительность: 10 эпох на первых четырех и 15 эпох на последних двух итерациях
- Результаты:
  - 1-я итерация
    - Уровень разреженности: 0.966
    - Точность сразу после удаления весов: 96%
    - Точность после файн-тюнинга: 96%
  - 2-я итерация
    - Уровень разреженности: 0.977
    - Точность сразу после удаления весов: 94%
    - Точность после файн-тюнинга: 96%
  - 3-я итерация
    - Уровень разреженности: 0.984
    - Точность сразу после удаления весов: 88%
    - Точность после файн-тюнинга: 95.5%
  - 4-я итерация
    - Уровень разреженности: 0.989
    - Точность сразу после удаления весов: 75%
    - Точность после файн-тюнинга: 95%
  - 5-я итерация
    - Уровень разреженности: 0.992
    - Точность сразу после удаления весов: 61%
    - Точность после файн-тюнинга: 95%
  - 6-я итерация

        - Уровень разреженности: 0.995
        - Точность сразу после удаления весов: 61%
        - Точность после файн-тюнинга: 94.5%

Итоговая точность после последней итерации составила 94.5%, что показывает, что постепенный прунинг с дообучением более эффективен для сохранения точности модели.

На датасете CIFAR100:

4.3.4 Однократный прунинг 95% весов с последующим дообучением

- Конфигурация:
    - Точность модели до прунинга 84%
    - Критерий отбора карт для удаления: L1 норма
    - Loss функция: кросс-энтропия
    - Оптимизатор: Adam
    - Learning rate: 3e-4, снижаемый в 10 раз на 19-й эпохе с помощью torch.optim.lr_scheduler.MultiStepLR
    - Weight decay: 5e-4
    - Длительность: 30 эпох
- Результаты:
    - Изначальная точность сразу после удаления весов оказалась на уровне случайного угадывания
    - Итоговая точность после файн-тюнинга составила 80%

4.3.5 Шесть итераций с удалением равной доли весов и дообучением на каждой итерации начиная с непрореженной сети до уровня разреженности 0.95

- Конфигурация:
    - Точность модели до прунинга 84%
    - Критерий отбора карт для удаления: L1 норма
    - Loss функция: кросс-энтропия
    - Оптимизатор: Adam
    - Learning rate: 3e-5
    - Weight decay: 5e-4
    - Длительность: 10 эпох на первых четырех и 15 эпох на последних двух итерациях
- Результаты:
    - 1-я итерация
        - Уровень разреженности: 0.39
        - Точность сразу после удаления весов: 83%

        - Точность после файн-тюнинга: 84%
    - 2-я итерация
        - Уровень разреженности: 0.63
        - Точность сразу после удаления весов: 74%
        - Точность после файн-тюнинга: 84%
    - 3-я итерация
        - Уровень разреженности: 0.77
        - Точность сразу после удаления весов: 56%
        - Точность после файн-тюнинга: 83.5%
    - 4-я итерация
        - Уровень разреженности: 0.86
        - Точность сразу после удаления весов: 28%
        - Точность после файн-тюнинга: 83%
    - 5-я итерация
        - Уровень разреженности: 0.92
        - Точность сразу после удаления весов: 13%
        - Точность после файн-тюнинга: 82%
    - 6-я итерация
        - Уровень разреженности: 0.95
        - Точность сразу после удаления весов: 4%
        - Точность после файн-тюнинга: 81%

4.3.6 Шесть итераций прунинга с уровня разреженности 0.77 до 0.95 с удалением равной доли весов и дообучением на каждой итерации

Чтобы увеличить точность прореженной до уровня 0.95 модели, было принято решение провести итеративный прунинг более плавно. Отправной точкой была выбрана модель с уровнем разреженности 0.77 и точностью 83.5%

- Конфигурация:
    - Точность модели до прунинга 83.5%
    - Критерий отбора карт для удаления: L1 норма
    - Loss функция: кросс-энтропия
    - Оптимизатор: Adam
    - Learning rate: 3e-5
    - Weight decay: 5e-4
    - Длительность: 10 эпох на первых четырех и 15 эпох на последних двух итерациях
- Результаты:
    - 1-я итерация

        - Уровень разреженности: 0.82
        - Точность сразу после удаления весов: 52%
        - Точность после файн-тюнинга: 83.5%
    - 2-я итерация
        - Уровень разреженности: 0.86
        - Точность сразу после удаления весов: 38.5%
        - Точность после файн-тюнинга: 83%
    - 3-я итерация
        - Уровень разреженности: 0.89
        - Точность сразу после удаления весов: 29%
        - Точность после файн-тюнинга: 83%
    - 4-я итерация
        - Уровень разреженности: 0.92
        - Точность сразу после удаления весов: 16%
        - Точность после файн-тюнинга: 82%
    - 5-я итерация
        - Уровень разреженности: 0.94
        - Точность сразу после удаления весов: 16%
        - Точность после файн-тюнинга: 82%
    - 6-я итерация
        - Уровень разреженности: 0.95
        - Точность сразу после удаления весов: 14.5%
        - Точность после файн-тюнинга: 82%

Итоговая точность после последней итерации составила 82%, что подтверждает эффективность более плавного итеративного подхода к прунингу.

4.3.7 Шесть итераций прунинга с уровня разреженности 0.95 до 0.993 с удалением равной доли весов и дообучением на каждой итерации

- Конфигурация:
    - Точность модели до прунинга 82%
    - Критерий отбора карт для удаления: L1 норма
    - Loss функция: кросс-энтропия
    - Оптимизатор: Adam
    - Learning rate: 3e-5
    - Weight decay: 5e-4
    - Длительность: 12 эпох на первых четырех и 16 эпох на последних двух итерациях
- Результаты:
    - 1-я итерация
        - Уровень разреженности: 0.965

        - Точность сразу после удаления весов: 7.5%
        - Точность после файн-тюнинга: 79%
    - 2-я итерация
        - Уровень разреженности: 0.975
        - Точность сразу после удаления весов: 10%
        - Точность после файн-тюнинга: 77.5%
    - 3-я итерация
        - Уровень разреженности: 0.982
        - Точность сразу после удаления весов: 3%
        - Точность после файн-тюнинга: 75%
    - 4-я итерация
        - Уровень разреженности: 0.987
        - Точность сразу после удаления весов: 7%
        - Точность после файн-тюнинга: 72.5%
    - 5-я итерация
        - Уровень разреженности: 0.99
        - Точность сразу после удаления весов: 4%
        - Точность после файн-тюнинга: 71%
    - 6-я итерация
        - Уровень разреженности: 0.993
        - Точность сразу после удаления весов: 3%
        - Точность после файн-тюнинга: 67%

Итоговая точность составила 67%, что показывает, что экстремальный уровень прунинга может существенно снизить точность модели.

4.3.8 Файн-тюнинг модели, прореженной до уровня 0.95 на CIFAR10 под датасет CIFAR100

Также был опробован метод обучения изначально прореженной модели. За основу была взята сеть с уровнем разреженности 0.95 и точностью 96% на CIFAR10, у которой линейный слой был заменен на 100 выходов и инициализирован из равномерного распределения Ксавье.

- Конфигурация:
    - Loss функция: кросс-энтропия
    - Оптимизатор: Adam
    - Learning rate: 1e-3, снижаемый в 10 раз на 20-й, 31-й и 46-й эпохах с помощью torch.optim.lr_scheduler.MultiStepLR
    - Weight decay: 5e-4
    - Длительность: 55 эпох
- Результаты:

- Точность модели после файн-тюнинга составила 77%, что показывает, что перенос знаний между датасетами и дообучение на новом датасете менее эффективны, чем прунинг на том же датасете.

5. Заключение

В рамках данной работы был проведен анализ различных техник неструктурированного прунинга на примере сверточной нейронной сети ResNet-34, использованной для задач классификации изображений на датасетах CIFAR10 и CIFAR100. На основе проведенных экспериментов можно сделать следующие выводы:

Эффективность прунинга на основе L1 нормы: Удаление весов на основе L1 нормы показало хорошие результаты. Данный метод продемонстрировал способность значительно уменьшить количество параметров модели при сохранении высокой точности.

Однократный прунинг: Однократное удаление весов с последующим дообучением позволило достигнуть удовлетворительных результатов. Например, на датасете CIFAR10, модель с 95% удаленных весов смогла восстановить точность до исходного уровня после дообучения. Однако, при более высокой степени разреженности (99.5%), качество модели значительно падало, несмотря на дообучение.

Итеративный прунинг: Итеративный прунинг с поэтапным удалением весов и последующим дообучением на каждом этапе показал лучшие результаты по сравнению с однократным прунингом. Этот метод позволил более плавно адаптировать модель к потере весов, что особенно важно при высоких уровнях разреженности. На датасете CIFAR10, итеративный прунинг до уровня разреженности 0.995 позволил сохранить точность на уровне 94.5%, а на CIFAR100 позволил удалить 82% весов с просадкой точности 0.5%.

Влияние уровня разреженности на производительность: На датасете CIFAR100, модель с исходной точностью 84% при однократном прунинге до уровня разреженности 0.95 смогла восстановить точность до 80% после дообучения. Более плавный итеративный прунинг позволил добиться лучшего результата (82%) при той же степени разреженности. Более высокая степень разреженности (свыше 95%) приводит к значительному снижению точности модели, даже при использовании итеративного подхода.

Таким образом, проведенные эксперименты подтверждают, что прунинг на основе L1 нормы в сочетании с итеративным дообучением является эффективным методом для оптимизации нейронных сетей, позволяющим значительно уменьшить количество параметров без существенной потери точности. Однако, для дальнейшего повышения эффективности и применения в более широком спектре задач, необходимы дополнительные исследования и усовершенствования существующих методов прунинга.

6. Список литературы

1. Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, Tijmen Blankevoort. 2021. A White Paper on Neural Network Quantization https://doi.org/10.48550/arXiv.2106.08295

2. Geoffrey Hinton, Oriol Vinyals, Jeff Dean. 2015. Distilling the Knowledge in a Neural Network https://doi.org/10.48550/arXiv.1503.02531

3. Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, Alexandra Peste. 2021. Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks https://doi.org/10.48550/arXiv.2102.00554

4. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. 2015. Deep Residual Learning for Image Recognition https://doi.org/10.48550/arXiv.1512.03385

5. Alex Krizhevsky. 2009. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

6. D Whitley, T Starkweather, C Bogart. 2003. Genetic algorithms and neural networks: optimizing connections and connectivity. Computer Science Department, Colorado State University, Fort Collins, CO 80523, USA https://doi.org/10.1016/0167-8191(90)90086-O

7. Song Han, Jeff Pool, John Tran, William J. Dally. 2015. Learning both Weights and Connections for Efficient Neural Networks https://papers.nips.cc/paper/2015/file/ae0eb3eed39d2bcef4622b2499a05fe6-Paper.pdf https://doi.org/10.48550/arXiv.1506.02626

8. Georg Thimm, Emile Fiesler. 1995. Evaluating Pruning Methods. Proceedings of the International Symposium on Artificial Neural Networks National Chiao-Tung University, Hsinchu, Taiwan, ROC, December 18-20 https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2bf84f883468d1a25481c949d0dc68ca186e5f24

9. Sian Jin, Sheng Di, Xin Liang, Jiannan Tian, Dingwen Tao, Franck Cappello. 2019. DeepSZ: A Novel Framework to Compress Deep Neural Networks by Using Error-Bounded Lossy Compression. HPDC '19: Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing. https://doi.org/10.1145/3307681.3326608

10. Yann Le Cun, John S. Denker, Sara A. Solla. 1989. Optimal Brain Damage

https://papers.nips.cc/paper/1989/file/6c9882bbac1c7093bd25041881277658-Paper.pdf

11. Petr Hurtik, Nicolás Madrid. 2015. Bilinear Interpolation over fuzzified images: Enlargement. IEEE International Conference on Fuzzy Systems At:

Istanbul.
https://www.researchgate.net/publication/280713578_Bilinear_Interpolation_over_fuzzified_images_Enlargement

12. Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, Quoc V. Le. 2019. AutoAugment: Learning Augmentation Policies from Data. https://doi.org/10.48550/arXiv.1805.09501

13. https://www.image-net.org

14. Xavier Glorot, Y. Bengio. 2010. Journal of Machine Learning Research 9:249-256.
https://www.researchgate.net/publication/215616968_Understanding_the_difficulty_of_training_deep_feedforward_neural_networks

15. Diederik P. Kingma, Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. 3rd International Conference for Learning Representations, San Diego, 2015. https://doi.org/10.48550/arXiv.1412.6980