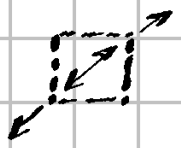
# EfficientNetV2: Smaller Models and Faster Training.

Более быстрая работа, чем у предыдущих моделей, скорость обучения и лучшая эффективность параметров.

Используется комбинация поиск + масштабирование, чтобы оптимизировать скорость обучения и эффективность параметров.

Модель EfficientNetV2 меньше остальных, но обучается в 6.8 раз быстрее.

- Размер изображения ↑ => скорость обучения ↑
  => точность ↓



Чтобы компенсировать ↓ точности => авторы создали метод, который адаптивно изменяет регуляризацию (например, увеличение данных) вместе с размером из-я.

## (1) Введение:

Предыдущая модель – EffecientNet. Исследование показало, что:

1) Обучение с очень большими размерами изображений происходит медленно
2) Свёртки по глубине происходят медленно на ранних слоях
3) Одинаковое масштабирование на каждом этапе не является оптимальным.

Чтобы решить проблемы выше авторы создали пространство поиска с доп. операциями FusedMBConv и применением NAS с учётом обучения и масштабирования для совместной оптимизации точности модели, ск. обучения и размера параметров. Такая модель обучается в 4 раза быстрее, при том, что она в 6.8 раз меньше по кол-ву параметров.

## ② EfficientNet V1

Повышение точности модели CNN связано с ↑ их размера и выч. сложности.

EfficientNet – семейство моделей, предложенных для баланса м/у точностью и эффективностью.

○ Проблема масштабирования НС.

• Традиционные методы масштабирования:

1) Масштабирование по глубине (Deeping) – увеличение кол-ва слоёв в сети.
2) Масштабирование по ширине (Widening) – увеличение кол-ва каналов в каждом слое.
3) масштабирование по разрешению (Resolution Scaling) – увеличение разрешения выходных изображений.

• Ограничения одностороннего масштабирования:

Масштабирование только по 1 из этих направлений может привести к неэффективному исп. ресурсов:

1) Только глубина: очень глубокие сети могут страдать от проблем с обучением, таких как исчезающий градиент.
* ситуация, когда значения градиента (скорость изм. ф-ции) → 0.
2) Только ширина: широкие сети требуют больше памяти и выч. ресурсов.
3) Только разрешение: высокое разрешение изображений ↑ выч. сложность и может привести к переобучению.

⇒ надо найти оптимальный путь.

## o EffecientNet V1: Составное масштабирование (Compaund Scaling).

• Основная идея:

EffecientNet V1 вводит составное масштабирование, которое одновременно масштабирует глубину, ширину и разрешение изображения сети с помощью одного коэф. масштабирования $\varphi$.

• Формула составного масштабирования.

- Глубина: $d = \alpha^{\varphi}$ (depth)
- Ширина: $w = \beta^{\varphi}$ (width)
- Разрешение: $r = \gamma^{\varphi}$ (resolution)

Здесь $\alpha, \beta, \gamma$ - коэф., определяющие насколько нужно масштабировать каждый параметр, а $\varphi$ - общий коэф. масштабирования, контролирующий общую нагрузку на выч. рес-ы.

• Ограничения на коэф-ы:

Коэффициенты выбираются таким образом, чтобы удовлетворять следующим условиям:

$$\begin{cases} \alpha \cdot \beta^2 \cdot \gamma \approx 2 \\ \alpha, \beta, \gamma \geq 1 \end{cases}$$

- Базовая модель EfficientNet

  - Использование нейроавтоматического поиска архитектуры (NAS)

Авторы использовали метод AutoML MNAS для поиска оптимальной базовой архитектуры сети, названной EfficientNet-B0. Этот подход учитывает как точность модели, так и её выч. эф-ть.

  - Особенности базовой архитектуры
    - MB Conv блоки: использованы мобильные свёртки с расширением (Mobile Inverted Bottleneck Convolution)
    - Swish активация: нелинейная ф-ция активации вместо ReLU, т.к. Swish менее подвержена "исч-ю град."
      Swish: $f(x) = x \cdot \text{sigmoid}(x)$, для более глубоких сл-в.
      ReLU: $f(x) = \max(0, x)$, для 1-ых слоёв.
    - Сжатие и возбуждение (Squeeze-and-Excitation): механизмы внимания для улучшения признаков.

- Применение составного масштабирования.

  - Создание семейства моделей EfficientNet:

Используя базовую EfficientNet-B0, применяя сост. масштабирование с разными значениями $\varphi$, были созданы модели от EfficientNet-B1 до EfficientNet-B7.

  - Пример масштабирования: EfficientNet-B1 → $\varphi = 1$
    EfficientNet-B2 → $\varphi = 2$
    ...
    EfficientNet-B7 → $\varphi = 7$.

## ③ EffecientNetV2

- Главное отличие – оптимизация для скорость обучения.

- Прогрессивное обучение (Progressive Learning)
  Эта техника состоит в том, что модель обучается сначала на данных с низким разрешением и постепенно увеличивает разрешение входов по мере того, как обучается на более поздних этапах. Это снимает выч. нагрузку на нач. этапах и помогает адаптироваться к сложным данным.

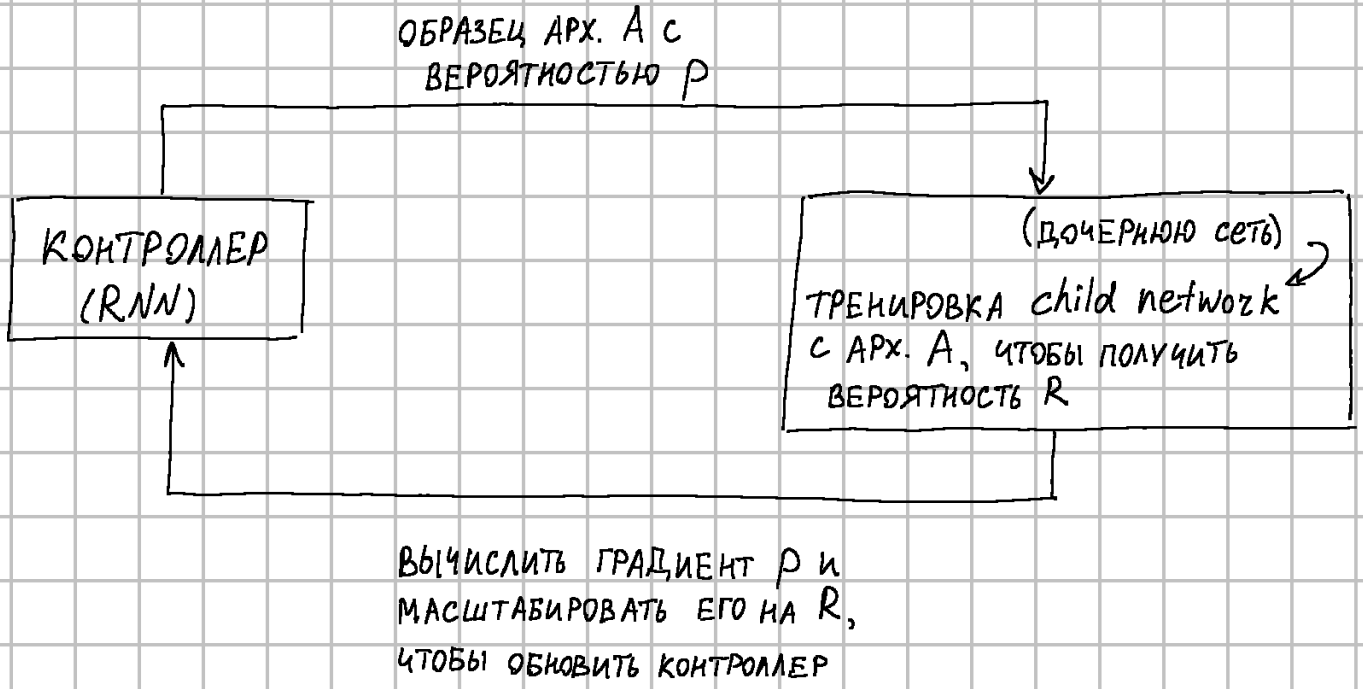
- Гибридные блоки (Fused MBConv)

EffecientNetV2 использует новый тип блоков, который называется Fused MBConv. Эти блоки сочетают в себе традиционные свёрточные операции и операции расширения (Expansion), что делает их более эффективными в плане вычислений по сравн-ю с классическими блоками MBConv, используемыми в 1-ой версии.
- Fused MBConv объединяет свёрточные и расширенные операции в один шаг, что ↓ выч. затраты и ↑ произв-ть.
- Fused MBConv эффективна на ранних этапах обучения (слои 1–3).

# Quoc V. Le:

2016: Neural Architecture Search with Reinforcement Learning (NAS RL)

Основная идея заключается в том, что есть контроллер. Выбирается архитектура и производится оценка во время обучения, а также происходит проверка на исх. (валидационном) наборе. Полученная точность влияет на вознаграждение контроллера, которое представляет собой точность валидации R.



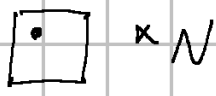

В итоге можно будет получить архитектуры с высокой точностью валидации.

## 2017: Learning Transferable Architectures for Scalable Image Recognition (LTAS IR)

Здесь говорится о переносе обучаемых архитектур для масштабируемого распознавания изображений.

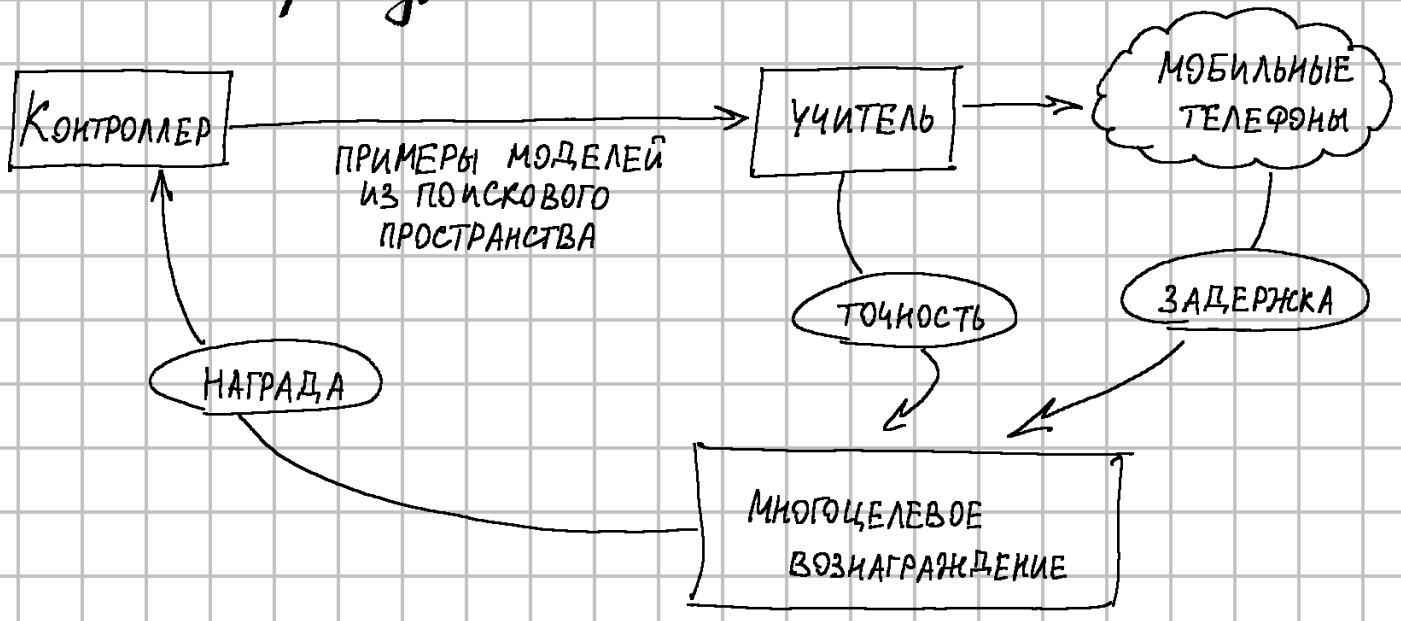
Авторы сузили область поиска, заметив, что большинство созданных вручную архитектур (например, моб. сети, RESNET) сводятся к созданию ячейки и её последующему многократному повторению.

=> авторы решили разработать "ячейку", чтобы повторять её N раз.

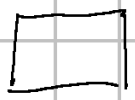



## 2018: MnasNet: Platform-Aware Neural Architecture Search for Mobile.

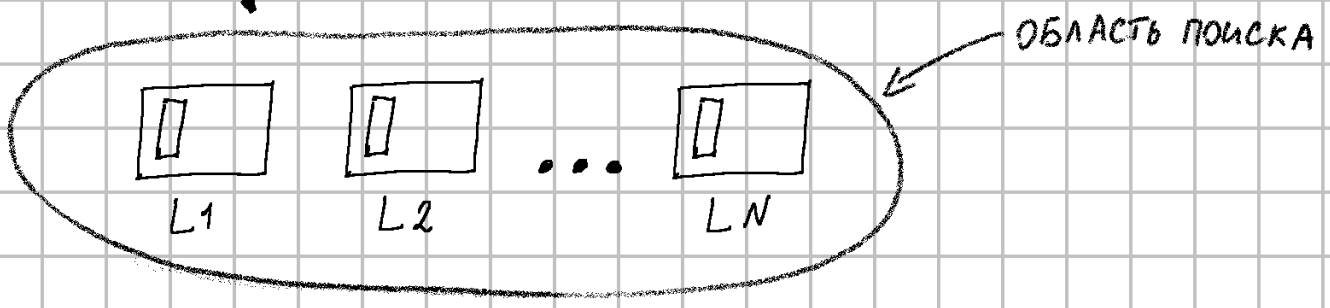
Авторы изменили обл. поиска:



Теперь есть несколько блоков (пусть их будет 7)




Выбираем 1-ый слой в каждом из блоков, а затем повторяем его несколько раз, что также явл. гиперпараметром, кот. мы ищем.




Также авторы включили задержку в конечную ф-цию вознаграждения => учитывается не только точность на проверочном наборе данных, но и задержка.

* Это сделало статью подходящей для поиска архитектур, кот. можно использовать на моб. устр-вах и др. периферийных устр-вах. Процесс поиска тот же, изм. только ф-ция вознаграждения. Тут не исп-ся ам. усиления, вместо него исп. PPO от OpenAI.

2019: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.

... + Mingxing Tan


Авторы взяли базовую архитектуру Во и добавили в неё параметры.

2020: Пандемия ...

2021: EfficientNet V2.