EfficientNetV2: Меньшие модели и более быстрое обучение

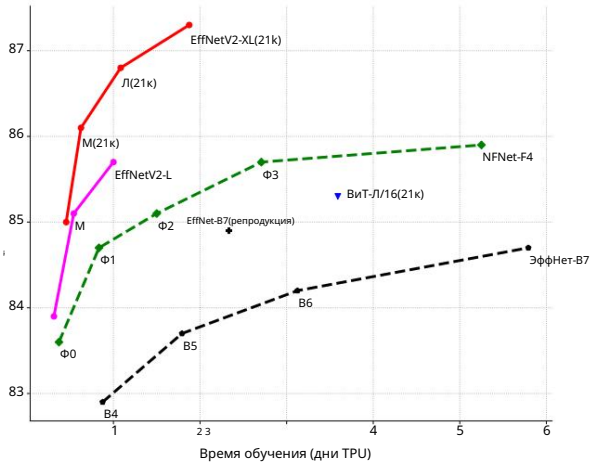
Минсин Тан [1] Куок В. Ле [1]

Абстрактный

В этой статье представлен EfficientNetV2, новое семейство сверточных сетей, которые имеют более быструю работу. скорость обучения и лучшая эффективность параметров чем предыдущие модели. Для разработки этих моделей, мы используем комбинацию поиска и масштабирования нейронной архитектуры с учетом обучения для совместной оптимизации Скорость обучения и эффективность параметров. Модели были найдены из поискового пространства, обогащенного с новыми операционными системами, такими как Fused-MBConv. Наши эксперименты показывают, что модели EfficientNetV2 обучаются гораздо быстрее, чем самые современные модели, при этом будучи в 6,8 раза меньше.

Наше обучение можно еще больше ускорить, постепенно увеличивая размер изображения во время обучения, но это часто приводит к снижению точности. Чтобы компенсировать это снижение точности, мы предлагаем усовершенствованный метод прогрессивного обучения, который адаптивно регулирует регуляризацию (например, увеличение данных) вместе с размером изображения.

Благодаря прогрессивному обучению наша EfficientNetV2 значительно превосходит предыдущие модели на наборах данных ImageNet и CIFAR/Cars/Flowers. предварительно обучаясь на том же ImageNet21k, наш Effi-cientNetV2 достигает 87,3% точности топ-1 на ImageNet ILSVRC2012, превосходящий недавний ViT на 2,0% точности при обучении 5x-11x быстрее, используя те же вычислительные ресурсы. Код доступен по адресу https://github.com/google/automl/tree/master/effectivenetv2.



(а) Эффективность обучения.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (наш)
Топ-1 Акк.	84,3%	84.0%	83,1%	83,9%
Параметры	43 млн.	164M	86M	24M

(б) Эффективность параметра.

Рисунок 1. ImageNet ILSVRC2012 top-1 Точность в сравнении с обучением Время и параметры – Модели, помеченные тегом 21k, предварительно обучены. на ImageNet21k, а другие обучаются непосредственно на ImageNet ILSVRC2012. Время обучения измеряется с 32 ядрами TPU. Все Модели EfficientNetV2 обучаются с помощью прогрессивного обучения. Наши EfficientNetV2 обучается в 5–11 раз быстрее других, используя при этом до В 6,8 раз меньше параметров. Подробности в таблице 7 и на рисунке 5.

1. Введение

Эффективность обучения важна для глубокого обучения как модели размер и размер обучающих данных становятся все больше. Например , GPT-3 (Brown et al., 2020), с гораздо большей моделью и больше данных обучения, демонстрирует замечательную способность к обучению с нескольких выстрелов, но это требует недель обучения с тысячами графических процессоров, что затрудняет переобучение или улучшать.

В последнее время все больший интерес вызывает эффективность обучения. Например, NFNets (Брок и др., 2021) нацелены на улучшение Эффективность обучения за счет устранения дорогостоящей пакетной нормализации ; Несколько недавних работ (Шринивас и др., 2021) фокусируются на повышении скорости обучения путем добавления слоев внимания в сверточные сети (ConvNets); Vision Transformers (Dosovitskiy et al., 2021) повышает эффективность обучения на больших наборах данных с использованием блоков Transformer. Однако эти методы часто сопровождаются большими накладными расходами на большой размер параметра, как показано на рисунке 1(б).

В этой статье мы используем комбинацию поиска нейронной архитектуры с учетом обучения (NAS) и масштабирования для улучшения обоих Скорость обучения и эффективность параметров. Учитывая параметр-

1Google Research, Brain Team. Адрес для переписки: Mingxing Тан <tanmingxing@google.com>.

Труды 38-й Международной конференции по машиностроению Обучение, PMLR 139, 2021.

Для повышения эффективности EfficientNets (Tan & Le, 2019a) мы начинаем с систематического изучения узких мест обучения в EfficientNets . Наше исследование показывает, что в EfficientNets: (1) обучение с очень большими размерами изображений происходит медленно; (2) свертки по глубине происходят медленно на ранних слоях. (3) одинаковое масштабирование на каждом этапе не является оптимальным. Основываясь на этих наблюдениях, мы проектируем пространство поиска, обогащенное дополнительными операциями, такими как Fused-MBConv, и применяем NAS с учетом обучения и масштабирование для совместной оптимизации точности модели, скорости обучения и размера параметров. Наши найденные сети, названные EfficientNetV2, обучаются до 4 раз быстрее, чем предыдущие модели (рисунок 3), при этом будучи до 6,8 раз меньше по размеру параметров.

Наше обучение может быть еще больше ускорено за счет постепенного увеличения размера изображения во время обучения. Во многих предыдущих работах, таких как прогрессивное изменение размера (Howard, 2018), FixRes (Touvron et al., 2019) и Mix&Match (Hoffer et al., 2019), использовались меньшие размеры изображений при обучении; однако они обычно сохраняют одинаковую регуляризацию для всех размеров изображений, что приводит к снижению точности. Мы утверждаем, что сохранение одинаковой регуляризации для разных размеров изображений не является идеальным вариантом: для одной и той же сети небольшой размер изображения приводит к малой емкости сети и, следовательно, требует слабой регуляризации; наоборот, большой размер изображения требует более сильной регуляризации для борьбы с переобучением (см. Раздел 4.1). Основываясь на этом понимании, мы предлагаем улучшенный метод прогрессивного обучения: в ранние эпохи обучения мы обучаем сеть с небольшим размером изображения и слабой регуляризацией (например, выпадением и увеличением данных), затем постепенно увеличиваем размер изображения и добавляем более сильную регуляризацию . Наш подход , основанный на прогрессивном изменении размера (Howard, 2018), но с динамической настройкой регуляризации, может ускорить обучение, не вызывая снижения точности.

Благодаря улучшенному прогрессивному обучению наш EfficientNetV2 достигает хороших результатов на наборах данных ImageNet, CIFAR-10, CIFAR- 100, Cars и Flowers. На ImageNet мы достигаем точности top-1 85,7%, при этом обучаясь в 3–9 раз быстрее и будучи в 6,8 раза меньше предыдущих моделей (рисунок 1). Наш EfficientNetV2 и прогрессивное обучение также облегчают обучение моделей на более крупных наборах данных. Например, ImageNet21k (Russakovsky et al., 2015) примерно в 10 раз больше ImageNet ILSVRC2012, но наш EfficientNetV2 может завершить обучение в течение двух дней, используя умеренные вычислительные ресурсы 32 ядер TPUv3. Благодаря предварительному обучению на общедоступной сети ImageNet21k наша сеть EfficientNetV2 достигает точности 87,3% на топ-1 сети ImageNet ILSVRC2012, превосходя недавнюю сеть ViT-L/16 по точности на 2,0%, при этом обучение происходит в 5–11 раз быстрее (рисунок 1).

Наш вклад состоит из трех частей:

- Мы представляем EfficientNetV2, новое семейство меньших и более быстрых моделей. Найденный нашим NAS с поддержкой обучения и масштабирования, EfficientNetV2 превосходит предыдущие модели как по скорости обучения, так и по эффективности параметров.
- Мы предлагаем усовершенствованный метод прогрессивного обучения. ing, который адаптивно регулирует регуляризацию вместе с размером изображения. Мы показываем, что это ускоряет обучение и одновременно повышает точность.
- Мы демонстрируем до 11 раз более высокую скорость обучения и до 6,8 раз более высокую эффективность параметров на наборах данных ImageNet, CIFAR, Cars и Flowers по сравнению с предыдущими разработками.

2. Сопутствующие работы

Обучение и эффективность параметров: многие работы, такие как DenseNet (Huang et al., 2017) и EfficientNet (Tan & Le, 2019a), фокусируются на эффективности параметров, стремясь достичь лучшей точности с меньшим количеством параметров. Некоторые более поздние работы направлены на улучшение скорости обучения или вывода вместо эффективности параметров. Например, RegNet (Radosavovic et al., 2020), ResNeSt (Zhang et al., 2020), TResNet (Ridnik et al., 2020) и EfficientNet-X (Li et al., 2021) фокусируются на скорости вывода GPU и/или TPU; NFNets (Brock et al., 2021) и BoTNets (Srinivas et al., 2021) фокусируются на улучшении скорости обучения. Однако их скорость обучения или вывода часто сопровождается ценой большего количества параметров. Целью данной статьи является значительное повышение скорости обучения и эффективности параметров по сравнению с предшествующим уровнем техники.

Прогрессивное обучение: В предыдущих работах предлагались различные виды прогрессивного обучения, которые динамически изменяют настройки обучения или сети для GAN (Karras et al., 2018), трансферного обучения (Karras et al., 2018), состязательного обучения (Yu et al., 2019) и языковых моделей (Press et al., 2021). Прогрессивное изменение размера (Howard, 2018) в основном связано с нашим подходом, который направлен на скорость обучения. Однако обычно сопряжено с потерей точности . Другая тесно связанная работа — Mix&Match (Hoffer et al., 2019), которая случайным образом выбирает разные размеры изображений для каждой партии. Как прогрессивное изменение размера, так и Mix&Match используют одну и ту же регуляризацию для всех размеров изображений, что приводит к снижению точности. В этой статье наше главное отличие заключается в адаптивной настройке регуляризации, чтобы мы могли улучшить как скорость обучения, так и точность. Наш подход также частично вдохновлен учебным планом (Бенджио и др., 2009), в котором учебные примеры располагаются от простого к сложному.

Наш подход также постепенно увеличивает сложность обучения за счет добавления большей регуляризации, но мы не выбираем обучающие примеры выборочно.

Поиск на основе нейронной архитектуры (NAS): Автоматизируя процесс проектирования сети, NAS использовался для оптимизации архитектуры сети для классификации изображений (Zoph et al., 2018), обнаружения объектов (Chen et al., 2019; Tan et al., 2020), сегментации (Liu et al., 2019), гиперпараметров (Dong et al., 2020) и других приложений (Elsken et al., 2019).

Предыдущие работы NAS в основном были сосредоточены на повышении эффективности FLOP (Tan & Le, 2019b;a) или эффективности вывода (Tan et al., 2019; Cai et al., 2019; Wu et al., 2019; Li et al., 2021).

В отличие от предыдущих работ, в этой статье для оптимизации обучения используется NAS.
и эффективность параметров.

3. Проектирование архитектуры EfficientNetV2

В этом разделе мы изучим узкие места обучения Efficient- Net (Tan & Le, 2019a) и представим нашу концепцию обучения с учетом NAS и масштабирование, а также модели EfficientNetV2.

3.1 Обзор EfficientNet

EfficientNet (Tan & Le, 2019a) — это семейство моделей, которые оптимизирован для FLOPs и эффективности параметров. Он использует NAS для поиска базовой версии EfficientNet-B0, которая имеет Лучший компромисс между точностью и FLOP. Базовая модель затем масштабируется с помощью комплексной стратегии масштабирования для получения семейство моделей B1-B7. В то время как последние работы утверждали, большой прирост скорости обучения или вывода, они часто хуже EfficientNet по параметрам и FLOP эффективность (таблица 1). В этой статье мы стремимся улучшить скорость обучения при сохранении эффективности параметров.

Таблица 1. EfficientNets имеют хорошие параметры и эффективность FLOP.

	Топ-1 Асс.	Параметры	FLOPs
EfficientNet-B6 (Тан и Ле, 2019a)	84,6%	43 млн	19Б
ResNet-RS-420 (Белло и др., 2021)	84,4%	192 млн	64Б
NFNet-F1 (Брок и др., 2021)	84,7%	133 млн	36Б

3.2 Понимание эффективности обучения

Мы изучаем узкие места обучения EfficientNet (Тан и Ле, 2019a), далее также называется EfficientNetV1, и несколько простые приемы для повышения скорости тренировок.

Обучение с очень большими размерами изображений происходит медленно: Как и было указано Как было отмечено в предыдущих работах (Радосавович и др., 2020), большой размер изображения EfficientNet приводит к значительному использованию памяти . Поскольку общий объем памяти на GPU/TPU фиксирован, нам необходимо обучать эти модели с меньшим размером партии, что радикально замедляет обучение. Простое улучшение — применить FixRes (Touvron et al., 2019), используя меньшее изображение размер для обучения, чем для вывода. Как показано в Таблице 2, Меньший размер изображения приводит к меньшему количеству вычислений и позволяет большой размер партии, и таким образом повышает скорость обучения на до 2,2x. Примечательно, что, как указано в (Touvron et al., 2020; Брок и др., 2021), используя меньший размер изображения для обучения также приводит к немного лучшей точности. Но в отличие от (Туврона и др., 2019) мы не настраиваем слои после обучения.

Таблица 2. Точность и производительность обучения EfficientNet-B6 для различных размеров пакетов и размеров изображений.

		TPUv3 imgs/sec/core		V100 imgs/sec/gpu
	Топ-1 Акк.	партия=32	партия=128	партия=12	партия=24
размер поезда=512	84,3%	42	OOM	29	OOM
размер поезда=380	84,6%	76	93	37	52

В разделе 4 мы рассмотрим более продвинутый вариант обучения. подход, заключающийся в постепенной корректировке размера изображения и регуляризации во время обучения.

Глубинные свертки медленные на ранних уровнях, но эффективные на более поздних этапах: Другое узкое место обучения Ef-ficientNet возникает из-за обширных глубинных сверток (Sifre, 2014). Глубинные свертки имеют меньше параметров и FLOPs, чем обычные свертки, но они часто не могут в полной мере использовать современные ускорители. В последнее время Fused-MBConv предложен в (Gupta & Tan, 2019) и позже использовано в (Гупта и Акин, 2020; Сюн и др., 2020; Ли и др., 2021) для более эффективного использования мобильных или серверных ускорителей. Он заменяет глубинный conv3x3 и расширение conv1x1 в MBConv (Сандлер и др., 2018; Тан и Ле, 2019a) с один регулярный conv3x3, как показано на рисунке 2. Чтобы систематически сравнивать эти два строительных блока, мы постепенно замените оригинальный MBConv в EfficientNet-B4 на Fused- MBConv (таблица 3). При применении на ранних стадиях 1-3, Fused- MBConv может улучшить скорость обучения с небольшими накладными расходами по параметрам и FLOP, но если мы заменим все блоки на Fused-MBConv (стадия 1-7), затем значительно увеличивается параметры и FLOPs, а также замедляя обучение. Нахождение правильной комбинации этих двух зданий блоки MBConv и Fused-MBConv, нетривиальны, что мотивирует нас использовать поиск на основе нейронной архитектуры для автоматического поиска лучшей комбинации.

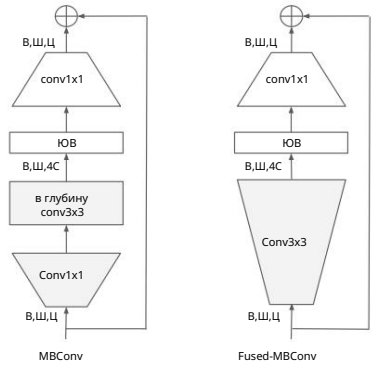


Рисунок 2. Структура MBConv и Fused-MBConv.

Таблица 3. Замена MBConv на Fused-MBConv. Без Fused обозначает, что все этапы используют MBConv, Fused stage1-3 обозначает замену MBConv на Fused-MBConv на этапе {2, 3, 4}.

	Параметры (М)	FLOPs (Б)	Топ-1 Acc.	ТПУ imgs/sec/core	V100 imgs/sec/gpu
Нет слитых	19.3	4.5	82.8%	262	155
Сплавленная стадия1-3	20.0	7.5	83.1%	362	216
Сплавленная стадия1-5	43.4	21,3	83,1%	327	223
Сплавленная стадия1-7	132.0	34,4	81,7%	254	206

Равномерное масштабирование каждого этапа не является оптимальным: Эффективно- чистое равномерно масштабирует все этапы, используя простое соединение Правило масштабирования. Например, когда коэффициент глубины равен 2, то все этапы в сетях удвоят количество слоев . Однако эти этапы не вносят равного вклада в

скорость обучения и эффективность параметров. В этой статье мы будем использовать стратегию неравномерного масштабирования для постепенного добавления больше слоев на более поздних этапах. Кроме того, EfficientNets агрессивно масштабирует размер изображения, что приводит к большой памяти потребление и медленное обучение. Чтобы решить эту проблему, мы немного изменить правило масштабирования и ограничить максимум размер изображения на меньшее значение.

3.3. NAS с поддержкой обучения и масштабирование

Для этого мы изучили несколько вариантов дизайна для улучшения скорости обучения. Для поиска лучших комбинаций Из этих вариантов мы теперь предлагаем NAS с поддержкой обучения.

Поиск NAS: наша структура NAS, ориентированная на обучение, в значительной степени основаны на предыдущих работах NAS (Tan et al., 2019; Tan & Le, 2019a), но нацелен на совместную оптимизацию точности, эффективность параметров и эффективность обучения на современных ускорителях . В частности, мы используем EfficientNet в качестве основы. Наше пространство поиска представляет собой поэтапно факторизованное пространство, похожее на (Tan et al., 2019), который состоит из вариантов дизайна для типы сверточных операций {MBConv, Fused-MBConv}, количество слоев, размер ядра {3x3, 5x5}, коэффициент расширения {1, 4, 6}. С другой стороны, мы уменьшаем размер пространства поиска путем (1) удаления ненужных параметров поиска, таких как объединение пропускать операции, так как они никогда не используются в исходном Efficient- Nets; (2) повторное использование тех же размеров каналов из магистрали поскольку они уже ищутся в (Tan & Le, 2019a). Поскольку пространство поиска меньше, мы можем применить обучение с подкреплением (Tan et al., 2019) или просто случайный поиск на многих более крупные сети, которые имеют сопоставимый размер с EfficientNet- B4. В частности, мы выбираем до 1000 моделей и обучаем каждая модель около 10 эпох с уменьшенным размером изображения для Обучение. Наше вознаграждение за поиск объединяет точность модели A, нормализованное время шага обучения S и размер параметра P, используя простое взвешенное произведение $A \cdot S^w \cdot P^v$, где $w = -0{,}07$ и $v = -0{,}05$ эмпирически определены для балансировки компромиссы, аналогичные (Тан и др., 2019).

Архитектура EfficientNetV2: Таблица 4 показывает архитектуру для нашей искомой модели EfficientNetV2-S. Сравнение к магистральной сети EfficientNet, наш поисковый EfficientNetV2 имеет несколько основных отличий: (1) Первое отличие заключается в том, что EfficientNetV2 широко использует как MBConv (Sandler и др., 2018; Тан и Ле, 2019a) и недавно добавленные fusion-MBConv (Gupta & Tan, 2019) в ранних слоях. (2) Во-вторых, EfficientNetV2 предпочитает меньший коэффициент расширения. для MBConv, поскольку меньшие коэффициенты расширения, как правило, имеют меньше накладных расходов на доступ к памяти. (3) В-третьих, EfficientNetV2 предпочитает меньшие размеры ядра 3x3, но добавляет больше слоев компенсировать уменьшенное рецептивное поле, возникшее в результате меньший размер ядра. (4) Наконец, EfficientNetV2 полностью удаляет последний этап шага-1 в исходной EfficientNet, возможно из-за большого размера параметров и доступа к памяти накладные расходы.

Таблица 4. Архитектура EfficientNetV2-S – MBConv и Fused-
Блоки MBConv описаны на рисунке 2.

Этап	Оператор	Шаг	#Каналы	#Слои
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3		24	2
2	Fused-MBConv4, k3x3	1	48	4
3	MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25		160	9
6 7	MBConv6, k3x3, SE0.25	2 1 2	256	15
	Conv1x1 и объединение и FC	-	1280	1

Масштабирование EfficientNetV2: Мы масштабируем EfficientNetV2-S до получить EfficientNetV2-M/L, используя аналогичное составное масштабирование как (Tan & Le, 2019a), с несколькими дополнительными оптимизациями: (1) мы ограничиваем максимальный размер выводимого изображения до 480, так как очень большие изображения часто приводят к дорогостоящему использованию памяти и накладные расходы на скорость обучения; (2) в качестве эвристики мы также постепенно добавить больше слоев на более поздние этапы (например, этап 5 и 6 в таблице 4) для увеличения пропускной способности сети без добавления большие накладные расходы времени выполнения.

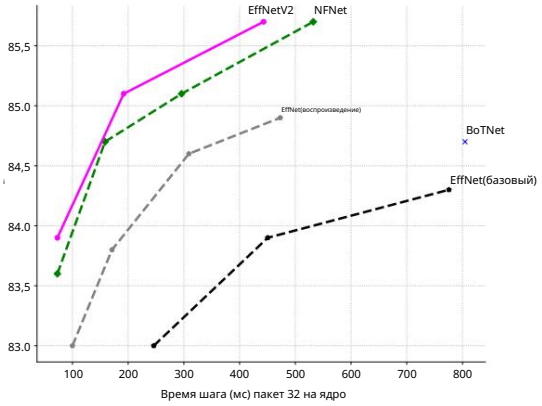


Рисунок 3. Точность ImageNet и время шага обучения на TPUv3 – Чем меньше время шага, тем лучше; все модели обучаются с фиксированным изображением размер без прогрессивного обучения.

Сравнение скорости обучения: На рисунке 3 сравнивается время шага обучения для нашей новой модели EfficientNetV2, где все модели обучаются с фиксированным размером изображения без прогрессивного обучения . Для EfficientNet (Tan & Le, 2019a) мы показываем два кривые: одна обучается с исходным размером вывода, а другой обучен с размером изображения примерно на 30% меньше, то же самое как EfficientNetV2 и NFNet (Туврон и др., 2019; Брок и др., 2021). Все модели обучаются с 350 эпохами, за исключением Сети NFNet обучаются в течение 360 эпох, поэтому все модели имеют аналогичное количество шагов обучения. Интересно, что мы наблюдаем что при правильном обучении EfficientNets все еще достигают довольно больших результатов сильный компромисс производительности. Что еще важнее, с нашими Предлагаемая нами модель NAS с поддержкой обучения и масштабирования Efficient- NetV2 обучается гораздо быстрее, чем другие недавние модели. Эти результаты также согласуются с нашими выводами, как показано в Таблице 7 и на Рисунке 5.

4. Прогрессивное обучение

4.1 Мотивация

Как обсуждалось в разделе 3, размер изображения играет важную роль в эффективности обучения. Помимо FixRes (Touvron et al., 2019), во многих других работах динамически изменяются размеры изображения во время обучения (Howard, 2018; Hoffer et al., 2019), но они часто приводят к снижению точности.

Мы предполагаем, что падение точности происходит из-за несбалансированной регуляризации: при обучении с разными размерами изображений мы также должны соответствующим образом корректировать силу регуляризации (вместо использования фиксированной регуляризации, как в предыдущих работах). Фактически, обычно большие модели требуют более сильной регуляризации для борьбы с переобучением: например, EfficientNet-B7 использует большее отсев и более сильное увеличение данных, чем B0. В этой статье мы утверждаем, что даже для той же сети меньший размер изображения приводит к меньшей емкости сети и, следовательно, требует более слабой регуляризации; наоборот , больший размер изображения приводит к большему количеству вычислений с большей емкостью и, следовательно, более уязвим для переобучения.

Для проверки нашей гипотезы мы обучаем модель, взятую из нашего поискового пространства, с различными размерами изображений и дополнениями данных (таблица 5). Когда размер изображения небольшой, наилучшая точность достигается при слабом дополнении; но для изображений большего размера она работает лучше при более сильном дополнении. Это понимание мотивирует нас адаптивно корректировать регуляризацию вместе с размером изображения во время обучения, что приводит к нашему улучшенному методу прогрессивного обучения.

Таблица 5. Точность ImageNet top-1. Мы используем RandAug (Cubuk et al., 2020) и сообщаем среднее значение и стандартное отклонение для 3 запусков.

	Размер=128	Размер=192	Размер=300
RandAug величина=5	78,3 ±0,16	81,2 ±0,06	82,5 ±0,05
RandAug величина=10	78,0 ±0,08	81,6 ±0,08	82,7 ±0,08
RandAug величина=15	77,7 ±0,15	81,5 ±0,05	83,2 ±0,09

4.2 Прогрессивное обучение с адаптивной регуляризацией

Рисунок 4 иллюстрирует процесс обучения нашего улучшенного прогрессивного обучения: в ранние эпохи обучения мы обучаем сеть с меньшими изображениями и слабой регуляризацией, так что сеть может легко и быстро изучать простые представления. Затем мы постепенно увеличиваем размер изображения, но также усложняем обучение, добавляя более сильную регуляризацию . Наш подход основан на (Howard, 2018) , который постепенно изменяет размер изображения, но здесь мы также адаптивно настраиваем регуляризацию.

Формально предположим, что все обучение состоит из N шагов, размер целевого изображения равен Se, со списком коэффициентов регуляризации. tude Φe = {φ, φ_e^k}, где k представляет собой тип регуляризации например, скорость отсева или значение скорости смешивания. Мы делим обучение на M этапов: для каждого этапа 1 i M модель обучается с размером изображения Si и величиной регуляризации

Рисунок 4. Процесс обучения в нашем улучшенном прогрессивном обучении. Он начинается с небольшого размера изображения и слабой регуляризации (эпоха = 1), а затем постепенно увеличивается сложность обучения с увеличением размера изображения и более сильной регуляризацией: больше процент отсева, величина RandAugment и коэффициент смешивания (например, эпоха = 300).

Φi = {φ φ_i^k}. Последний этап M будет использовать целевое изображение размер Se и регуляризация Φe. Для простоты мы эвристически выбираем начальный размер изображения S0 и регуляризацию Φ0, а затем используем линейную интерполяцию для определения значения для каждого этапа. Алгоритм 1 суммирует процедуру.

В начале каждого этапа сеть унаследует все веса из предыдущего этапа. В отличие от трансформаторов, чьи веса (например, встраивание позиции) могут зависеть от длины входных данных, веса ConvNet не зависят от размеров изображений и, таким образом, могут быть легко унаследованы.

Алгоритм 1. Прогрессивное обучение с адаптивной регуляризацией.

Входные данные: начальный размер изображения S0 и регуляризация $\{\varphi_0^k\}$.
Входные данные: Окончательный размер изображения Se и регуляризация $\{\varphi_e^k\}$.
Входные данные: Общее количество шагов обучения N и этапов M. для i = 0 до M 1 do
  Размер изображения: $S_i = S_0 + (S_e - S_0) \cdot \frac{i}{M-1}$
  Регуляризация: $R_i = \{\varphi_i^k = \varphi_0^k + (\varphi_e^k - \varphi_0^k) \cdot \frac{i}{M-1}\}$
  Обучите модель для $\frac{N}{M}$ шаги с Si и Ри.
конец для

Наше улучшенное прогрессивное обучение в целом совместимо с существующей регуляризацией. Для простоты в этой статье в основном изучаются следующие три типа регуляризации:

- Dropout (Srivastava et al., 2014): регуляризация на уровне сети , которая уменьшает коадаптацию путем случайного удаления каналов. Мы отрегулируем скорость удаления γ. •
- RandAugment (Cubuk et al., 2020): аугментация данных для каждого изображения с регулируемой величиной.
- Mixup (Zhang et al., 2018): кросс-изображение данных аугментации. Учитывая два изображения с метками (xi , yi) и (xj , yj), он объединяет их с коэффициентом смешивания λ: $\tilde{x}_i = \lambda x_j + (1 - \lambda) x_i$ и $\tilde{y}_i = \lambda y_j + (1 - \lambda) y_i$. Мы бы отрегулировали коэффициент смешивания λ во время обучения.

5. Основные результаты

В этом разделе представлены наши экспериментальные установки, основные результаты на ImageNet и результаты переноса обучения на CIFAR-10, CIFAR-100, Cars и Flowers.

5.1.ImageNet ILSVRC2012

Настройка: ImageNet ILSVRC2012 (Русаковский и др., 2015) содержит около 1,28 млн обучающих изображений и 50 000 проверочных изображения с 1000 классами. Во время поиска архитектуры или настройка гиперпараметров, мы резервируем 25 000 изображений (около 2%) из обучающего набора как минивал для оценки точности . Мы также используем минивал для выполнения ранней остановки. Наш Настройки обучения ImageNet в основном соответствуют EfficientNets (Tan & Le, 2019a): оптимизатор RMSProp с затуханием 0,9 и импульс 0,9; импульс нормы партии 0,99; спад веса 1e-5. Каждая модель обучается в течение 350 эпох с общим Размер партии 4096. Скорость обучения сначала разогревается от 0 до 0,256, а затем уменьшался на 0,97 каждые 2,4 эпохи. Мы использовать экспоненциальное скользящее среднее с коэффициентом затухания 0,9999, RandAugment (Cubuk et al., 2020), Mixup (Чжан и др., 2018), Dropout (Шривастава и др., 2014) и стохастический глубина (Хуан и др., 2016) с вероятностью выживания 0,8.

Таблица 6. Настройки прогрессивного обучения для EfficientNetV2.

	С		М		Л
	мин	макс	мин	макс	мин макс
Размер изображения	128	300	128	380	128 380
RandAugment	5	5	0	0	0,3 0,1 20 25
Mixup alpha	0	0.1			5 0,2 0.4
Коэффициент выпадения					0 0.1 0,4 0,5

Для прогрессивного обучения мы разделяем процесс обучения на четыре этапа, каждый из которых включает около 87 эпох: ранний этап использует небольшой размер изображения со слабой регуляризацией, в то время как на более поздних этапах используются изображения большего размера с более сильным регуляризация, как описано в алгоритме 1. Таблица 6 показывает минимум (для первого этапа) и максимум (для последний этап) значения размера изображения и регуляризации. Для простота, все модели используют одинаковые минимальные значения размера и регуляризация, но они принимают разные максимальные значения, поскольку более крупные модели обычно требуют большей регуляризации борьба с переобучением. После (Touvron et al., 2020) наши максимальный размер изображения для обучения примерно на 20% меньше вывод, но мы не настраиваем слои после обучения.

Результаты: Как показано в таблице 7, наши модели EfficientNetV2 значительно быстрее и достигают лучшей точности и параметр эффективности, чем предыдущие ConvNets и Transformers на ImageNet. В частности, наш EfficientNetV2-M достигает сопоставимой точности с EfficientNet-B7, в то время как Обучение в 11 раз быстрее с использованием тех же вычислительных ресурсов. Наш Модели EfficientNetV2 также значительно превосходят все последние модели RegNet и ResNeSt как по точности, так и по выводам. скорость. Рисунок 1 дополнительно визуализирует сравнение скорости обучения и эффективности параметров. Примечательно, что это ускорение сочетание прогрессивного обучения и лучших сетей, и мы изучим индивидуальное воздействие каждого из них в наши исследования абляции.

Недавно Vision Transformers продемонстрировали впечатляющие результаты. sive результаты по точности ImageNet и скорости обучения. Однако , здесь мы показываем, что правильно спроектированные ConvNets с усовершенствованный метод обучения все еще может значительно превзойти визуальные трансформаторы как по точности, так и по эффективности обучения. В частности, наш EfficientNetV2-L достигает 85,7% топ-1 точность, превосходящая ViT-L/16(21k), гораздо более крупную модель трансформатора, предварительно обученную на большем наборе данных ImageNet21k. Здесь ViT не очень хорошо настроены на ImageNet ILSVRC2012; DeiT используют ту же архитектуру, что и ViT, но достигают лучших результатов результаты за счет добавления большей регуляризации.

Хотя наши модели EfficientNetV2 оптимизированы для обучения , они также хорошо подходят для вывода, поскольку обучение скорость часто коррелирует со скоростью вывода. Рисунок 5 визуализирует размер модели, FLOPs и задержку вывода на основе в Таблице 7. Поскольку задержка часто зависит от оборудования и программное обеспечение, здесь мы используем те же модели изображений PyTorch кодовая база (Wightman, 2021) и запустить все модели на одном и том же машина с размером партии 16. В целом, наши модели имеют немного лучшие параметры/эффективность FLOP, чем Effi- cientNets, но наша задержка вывода до 3 раз быстрее, чем EfficientNets. По сравнению с последними ResNeSt, которые специально оптимизированы для графических процессоров, наш EfficientNetV2-M достигает Точность выше на 0,6%, а скорость вывода в 2,8 раза выше.

5.2.ImageNet21k

Настройка: ImageNet21k (Русаковский и др., 2015) содержит около 13 млн учебных изображений с 21 841 классами. Оригинал ImageNet21k не имеет разделения на обучение/оценку, поэтому мы резервируем 100 000 случайно выбранных изображений в качестве проверочного набора и используем оставаясь в качестве обучающего набора. Мы в основном повторно используем тот же обучающий набор настройки как у ImageNet ILSVRC2012 с некоторыми изменениями: (1) мы изменяем эпохи обучения на 60 или 30, чтобы сократить время обучения время и использовать косинусный спад скорости обучения, который может адаптироваться к различные шаги без дополнительной настройки; (2) поскольку каждое изображение имеет несколько меток, мы нормализуем метки, чтобы получить сумму 1 перед вычислением softmax loss. После предварительной тренировки на ImageNet21k, каждая модель точно настроена на ILSVRC2012 для 15 эпох с использованием косинусного спада скорости обучения.

Результаты: Таблица 7 показывает сравнение производительности. где модели, помеченные 21k, предварительно обучены на ImageNet21k и отлажены на ImageNet ILSVRC2012. По сравнению с недавним ViT-L/16(21k), наш EfficientNetV2-L(21k) улучшает точность топ-1 на 1,5% (85,3% против 86,8%), используя в 2,5 раза меньше параметров и в 3,6 раза меньше FLOP, при этом обучение и выводы выполняются в 6–7 раз быстрее.

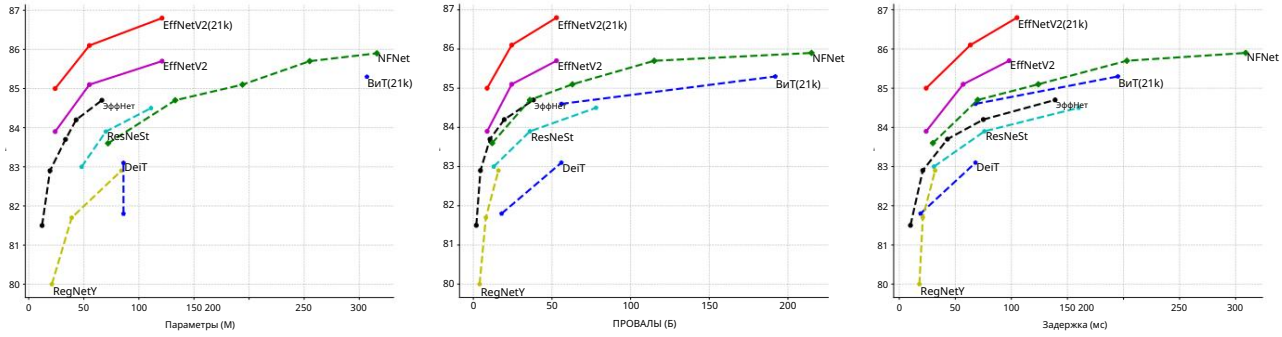
Мы хотели бы выделить несколько интересных наблюдений:

- Масштабирование размера данных более эффективно, чем простое масштабирование размера модели в режиме высокой точности: когда Точность топ-1 превышает 85%, очень сложно еще больше улучшите его, просто увеличив размер модели

Таблица 7. Результаты производительности EfficientNetV2 на ImageNet (Russakovsky et al., 2015) – время вывода измерялось на V100 GPU FP16 с размером пакета 16 с использованием той же кодовой базы (Wightman, 2021); Train-time — это общее время обучения, нормализованное для 32 ядер TPU. Модели, отмеченные как 21k, предварительно обучены на ImageNet21k с 13M изображениями, а другие обучены напрямую на ImageNet ILSVRC2012. с 1,28 млн изображений с нуля. Все модели EfficientNetV2 обучаются с помощью нашего улучшенного метода прогрессивного обучения.

	Модель	Топ-1 Acc.	Параметры	FLOPs	Время вывода (мс)	Время обучения (часы)
ConvNets и гибрид	EfficientNet-B3 (Тан и Ле, 2019a)	81,5%	12 млн	1,9 млрд	82,9% 19	10
	EfficientNet-B4 (Тан и Ле, 2019a)	млн	4,2 млрд	83,7%	30 млн 10	21
	EfficientNet-B5 (Тан и Ле, 2019a)	млрд	84,3%	43 млн	19 млрд	43
	EfficientNet-B6 (Тан и Ле, 2019a)	84,7%	66 млн		97	75
	EfficientNet-B7 (Тан и Ле, 2019a)			38Б	170	139
	RegNetY-8GF (Радосавович и др., 2020)	81,7%	39 млн.	8Б	21	-
	RegNetY-16GF (Радосавович и др., 2020)	82,9%	84М	83,0% 16Б	32	-
	ResNeSt-101 (Чжан и др., 2020)	48М	83,9%	70М 13Б	31	-
	ResNeSt-200 (Чжан и др., 2020)	84,5%	111М	83,8% 36Б	76	-
	ResNeSt-269 (Чжан и др., 2020)	56М	84,3%	78М 78Б	160	-
	TResNet-L (Ридник и др., 2020)	84,7%	73М	91Б -	45	-
	TResNet-XL (Ридник и др., 2020)	83,6%	72М	12Б -	66	-
	EfficientNet-X (Ли и др., 2021 г.)	84,7%	133М	36Б 85,1% 194М	-	-
	NFNet-F0 (Брок и др., 2021)	63Б 85.	7%	255М 115Б 85,9%	30	8.9
	NFNet-F1 (Брок и др., 2021)	316М	215Б		70	20
	NFNet-F2 (Брок и др., 2021)				124	36
	NFNet-F3 (Брок и др., 2021)				203	65
	NFNet-F4 (Брок и др., 2021)				309	126
	LambdaResNet-420-гибрид (Белло, 2021)	84,9%	125 млн.	-	-	67
	Гибрид BotNet-T7 (Сринивас и др., 2021 г.)	84,7%	75 млн	46 млрд 85,2%	-	95
	БиТ-М-Р152х2 (21к) (Колесников и др., 2020)	236 млн	135 млрд		500	-
Зрение Трансформеры	ВиТ-Б/32 (Досовицкий и др., 2021)	73,4%	88 млн	13 млрд 74,9% 87	13	-
	ВиТ-Б/16 (Досовицкий и др., 2021)	млн	56 млрд	81,8% 86 млн 18	68	-
	DeiT-B (ViT+reg) (Туврон и др., 2021)	млрд	83,1%	86 млн 56 млрд	19	-
	ДеиТ-Б-384 (ВиТ+рег) (Туврон и др., 2021)	81,4%	39 млн	8,4 млрд 82,2% 64	68	-
	Т2Т-ВиТ-19 (Юань и др., 2021)	млн	13 млрд	56 млрд	-	-
	Т2Т-ВиТ-24 (Юань и др., 2021)				-	-
	ВиТ-Б/16 (21к) (Досовицкий и др., 2021)	84,6%	87 млн.		68	-
	ВиТ-Л/16 (21к) (Досовицкий и др., 2021)	85,3%	304 млн	192 млрд	195	172
ConvNets (наш)	ЭффективныйNetV2-S	83,9%	22 млн	8,8 млрд 85,1% 54	24	7.1
	Эффективная сеть V2-M	млн	24 млрд	85,7% 120 млн 53	57	13
	ЭффективныйNetV2-L	млрд	84,9%	22 млн 8,8 млрд	98	24
	EfficientNetV2-S (21 тыс.)	86,2%	54 млн	24 млрд 86,8%	24	9.0
	EfficientNetV2-M (21 тыс.)	120 млн	53 млрд	87,3% 208 млн	57	15
	EfficientNetV2-L (21 тыс.)	94 млрд			98	26
	EfficientNetV2-XL (21 тыс.)				-	45

Мы не включаем модели, предварительно обученные на закрытых изображениях Instagram/JFT, а также модели с дополнительной обработкой или ансамблем.



(а) Параметры (б) Провалы (с) Задержка графического процессора V100 (пакет 16)

Рисунок 5. Размер модели, FLOPs и задержка вывода . Задержка измеряется с размером пакета 16 на графическом процессоре V100. 21k обозначает предварительно обученный на изображениях ImageNet21k, другие просто обучаются на ImageNet ILSVRC2012. Наш EfficientNetV2 имеет немного лучшую эффективность параметров с EfficientNet, но работает в 3 раза быстрее для вывода.

Таблица 8. Сравнение эффективности переноса обучения . Все модели предварительно обучены на ImageNet ILSVRC2012 и настроены на
Наборы данных ниже по течению. Точность переноса обучения усредняется по пяти запускам.

	Модель	Параметры ImageNet Acc.	CIFAR-10	CIFAR-100	Цветы	Автомобили
ConvNets	GPipe (Хуан и др., 2019)	556M 84,4	99,0	91,3	98,8	94,7
	EfficientNet-B7 (Тан и Ле, 2019a)	66M 84,7	98,9	91,7	98,8	94,7
Зрение Трансформеры	ViT-B/32 (Досовицкий и др., 2021)	88M 73,4	97,8	86,3	85,4	-
	ViT-B/16 (Досовицкий и др., 2021)	87M 74,9	98,1	87,1	89,5	-
	ViT-L/32 (Досовицкий и др., 2021)	306M 71,2	97,9	87,1	86,4	-
	ViT-L/16 (Досовицкий и др., 2021)	306M 76,5	97,9	86,4	89,7	-
	DeiT-B (ViT+регуляризация) (Туврон и др., 2021)	86M 81,8	99,1	90,8	98,4	92.1
	DeiT-B-384 (ViT+регуляризация) (Туврон и др., 2021) 86M	83,1	99,1	90,8	98,5	93.3
ConvNets (наш)	EfficientNetV2-S	24M 83,2	98,7±0,04	91,5±0,11	97,9±0,13	93,8±0,11
	Эффективная сеть V2-M	55M 85,1	99,0±0,08	92,2±0,08	98,5±0,08	94,6±0,10
	EfficientNetV2-L	121M 85,7	99,1±0,03	92,3±0,13	98,8±0,05	95,1±0,10

из-за сильного переобучения. Однако дополнительная предварительная подготовка Im-ageNet21K может значительно улучшить точность. Эффективность больших наборов данных также наблюдалась в предыдущих работах (Mahajan et al., 2018; Xie и др., 2020; Досовицкий и др., 2021).

- Предварительное обучение на ImageNet21k может быть весьма эффективным. Хотя ImageNet21k имеет в 10 раз больше данных, наше обучение Подход позволяет нам завершить предварительную подготовку Effi-cientNetV2 в течение двух дней с использованием 32 ядер TPU (вместо недель для ViT (Dosovitskiy et al., 2021)). Это эффективнее, чем обучение более крупных моделей на ImageNet . Мы предлагаем в будущих исследованиях крупномасштабных моделей использовать общедоступный ImageNet21k в качестве набора данных по умолчанию.

5.3 Передача наборов данных обучения

Настройка: Мы оцениваем наши модели на основе четырех методов трансферного обучения.
Наборы данных: CIFAR-10, CIFAR-100, Цветы и автомобили. Таблица 9 включает статистику этих наборов данных.

Таблица 9. Наборы данных трансферного обучения.

	Образы поездов	Образы оценок	Классы
CIFAR-10 (Крижевский и Хинтон, 2009)	50 000	10 000	10
CIFAR-100 (Крижевский и Хинтон, 2009)	50 000	10 000	100
Цветы (Нильсбак и Зиссерман, 2008)	2 040	6 149	102
Автомобили (Краузе и др., 2013)	8 144	8 041	196

Для этого эксперимента мы используем контрольные точки, обученные на ImageNet ILSVRC2012. Для честного сравнения, нет ImageNet21k
Здесь используются изображения. Наши настройки тонкой настройки в основном то же самое, что и обучение ImageNet с некоторыми модификациями, похожими (Досовицкий и др., 2021; Туврон и др., 2021): Мы используем меньший размер партии 512, меньшая начальная скорость обучения 0,001 с косинусным распадом. Для всех наборов данных мы обучаем каждую модель для фиксированные 10000 шагов. Поскольку каждая модель настроена очень точно несколько шагов, мы отключаем снижение веса и используем простой вырез дополнение данных.

Результаты: Таблица 8 сравнивает производительность переноса обучения. В целом, наши модели превосходят предыдущие Con- vNets и Vision Transformers для всех этих наборов данных, иногда с нетривиальной разницей: например, на CIFAR-100, EfficientNetV2-L обеспечивает точность на 0,6% выше, чем раньше GPipe/EfficientNets и точность на 1,5% выше, чем раньше Модели ViT/DeiT. Эти результаты показывают, что наши модели также обобщать далеко за пределы ImageNet.

6. Исследования абляции

6.1 Сравнение с EfficientNet

В этом разделе мы сравним наш EfficientNetV2 (V2 для сокращенно) с EfficientNets (Tan & Le, 2019a) (V1 сокращенно) при тех же настройках обучения и вывода.

Производительность при той же тренировке: Таблица 10 показывает Сравнение производительности с использованием тех же прогрессивных настроек обучения. Поскольку мы применяем то же самое прогрессивное обучение EfficientNet, скорость его обучения (снижена со 139 ч до 54ч) и точность (улучшение с 84,7% до 85,0%) лучше, чем в оригинальной статье (Tan & Le, 2019a). Однако , как показано в Таблице 10, наши модели EfficientNetV2 по-прежнему превзошли EfficientNets с большим отрывом: EfficientNetV2-M уменьшает параметры на 17% и FLOP на 37%, в то время как работает в 4,1 раза быстрее при обучении и в 3,1 раза быстрее при выводе чем EfficientNet-B7. Поскольку мы используем ту же самую учебную настройки здесь, мы приписываем выигрыш EfficientNetV2 архитектура.

Таблица 10. Сравнение с теми же настройками обучения – Наш новый
EfficientNetV2-M работает быстрее с меньшим количеством параметров.

	Acc. (%)	Params (M)	FLOPs (B)	TrainTime (час)	InferTime (PC)
V1-B7	85.0	66	38	54	170
V2-M (наш)	85,1	55 (-17%)	24 (-37%)	13 (-76%)	57 (-66%)

Масштабирование вниз: Предыдущие разделы в основном посвящены крупномасштабным моделям. Здесь мы сравниваем меньшие модели путем масштабирования вниз по нашему EfficientNetV2-S с использованием соединения EfficientNet масштабирование. Для удобства сравнения все модели обучаются без Прогрессивное обучение. По сравнению с EfficientNets небольшого размера (V1), наши новые модели EfficientNetV2 (V2) в целом быстрее при сохранении сопоставимой эффективности параметров.

Таблица 11. Уменьшение размера модели – Мы измеряем вывод
Пропускная способность (изображений/сек) на графическом процессоре V100 FP16 с размером пакета 128.

	Топ-1 Acc.	Параметры	FLOPs	Пропускная способность
B1-B1	79,0%	7.8M	0.7Б	2675
V2-B0	78,7%	7.4M	0,7Б (2,1x)	5739
B1-B2	79,8%	9.1M	1.0В	2003
B2-B1	79,8%	8.1M	1.2Б (2.0x)	3983
B1-B4	82,9%	19M	4.2В	628
V2-B3	82,1%	14M	3.0В (2.7x)	1693
B1-B5	83,7%	30M	9.9В	291
V2-S	83,6%	24M	8.8Б (3.1x)	901

6.2 Прогрессивное обучение для различных сетей

Мы снижаем эффективность нашего прогрессивного обучения
разные сети. Таблица 12 показывает сравнение производительности между нашим
прогрессивным обучением и базовым уровнем
обучение с использованием тех же моделей ResNet и EfficientNet.
Здесь базовые ResNets имеют более высокую точность, чем
оригинальная статья (He et al., 2016), поскольку они обучены
наши улучшенные настройки обучения (см. Раздел 5) с использованием большего количества
эпохи и лучшие оптимизаторы. Мы также увеличиваем изображение
размер от 224 до 380 для ResNets для дальнейшего увеличения
пропускная способность и точность сети.

Таблица 12. Прогрессивное обучение для ResNets и EfficientNets –
(224) и (380) обозначают размер выводимого изображения. Наше прогрессивное
обучение улучшает как точность, так и время обучения для всех сетей.

	Базовый		прогрессивный	
	Acc.(%)	TrainTime	Acc.(%)	TrainTime
ResNet50 (224)	78,1	4,9 ч	78,4	3,5 ч (-29%)
ResNet50 (380)	80,0	14,3 ч	80,3	5,8ч (-59%)
ResNet152 (380)	82,4	15,5 ч	82,9	7,2ч (-54%)
Эффективная сеть-B4	82,9	20.8ч	83,1	9,4ч (-55%)
Эффективная сеть-B5	83,7	42.9ч	84,0	15,2ч (-65%)

Как показано в Таблице 12, наше прогрессивное обучение в целом
сокращает время обучения и в то же время улучшает
точность для всех различных сетей. Неудивительно, что когда
размер изображения по умолчанию очень мал, например ResNet50(224) с
размер 224x224, ускорение обучения ограничено (ускорение 1,4x);
Однако, когда размер изображения по умолчанию больше, а модель
более сложен, наш подход достигает большего прироста точности и
эффективности обучения: для ResNet152(380) наш подход улучшает
скорость обучения в 2,1 раза с небольшим
более высокая точность; для EfficientNet-B4 наш подход улучшает
ускорить обучение в 2,2 раза.

6.3 Важность адаптивной регуляризации

Ключевым моментом нашего подхода к обучению является адаптивность
регуляризация, которая динамически корректирует регуляризацию
в соответствии с размером изображения. В этой статье выбран простой
прогрессивный подход из-за его простоты, но это также общий подход
Метод, который можно комбинировать с другими подходами.

Таблица 13 изучает нашу адаптивную регуляризацию на двух обучающих моделях.
настройки: один из них — постепенное увеличение размера изображения от
от малого к большому (Говард, 2018), а другой — случайным образом
выборка разного размера изображения для каждой партии (Хоффер и др.,
2019). Поскольку TPU необходимо перекомпилировать график для каждого
новый размер, здесь мы случайным образом выбираем размер изображения каждые восемь
эпохи вместо каждой партии. По сравнению с ванилью
подходы прогрессивного или случайного изменения размера, которые используют
та же регуляризация для всех размеров изображения, наша
адаптивная регуляризация повышает точность на 0,7%. Рисунок 6 далее
сравнивает кривую обучения для прогрессивного подхода.
Наша адаптивная регуляризация использует гораздо меньшую регуляризацию
для небольших изображений на ранних этапах обучения, что позволяет
модели быстрее сходятся и достигают лучшей конечной точности.

Таблица 13. Адаптивная регуляризация – Сравниваем ImageNet top-1
точность основана на среднем значении трех запусков.

	Ваниль	+наш адаптивный рег
Прогрессивное изменение размера (Говард, 2018)	84,3±0,14	85,1±0,07 (+0,8)
Случайное изменение размера (Хоффер и др., 2019)	83,5±0,11	84,2±0,10 (+0,7)

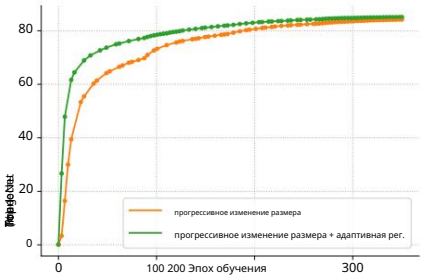


Рисунок 6. Сравнение кривых обучения . Наша адаптивная регуляризация
сходится быстрее и обеспечивает лучшую конечную точность.

7. Заключение

В этой статье представлен EfficientNetV2, новое семейство меньших
и более быстрые нейронные сети для распознавания изображений. Оптимизированный
Благодаря NAS с поддержкой обучения и масштабированию моделей наша Efficient-
NetV2 значительно превосходит предыдущие модели, в то время как
быть намного быстрее и эффективнее по параметрам. Для
дальнейшего ускорения обучения мы предлагаем усовершенствованный метод
прогрессивного обучения, которое совместно увеличивает размер изображения
и регуляризация во время обучения. Обширные эксперименты
показать, что наш EfficientNetV2 достигает хороших результатов на ImageNet и
CIFAR/Flowers/Cars. По сравнению с EfficientNet
и более поздние работы, наш EfficientNetV2 обучает до 11x
быстрее, при этом в 6,8 раза меньше.

Благодарности

Особая благодарность Лукасу Слоуну за помощь в открытии исходного кода.
Мы благодарим Руомина Пана, Шэн Ли, Эндрю Ли, Ханьсяо.
Лю, Цзиханг Дай, Нил Хоулсби, Росс Уайтман, Джереми
Ховард, Тхан Луонг, Дайи Пэн, Ифэн Лу, Да Хуан,
Чэнь Лян, Аравинд Шринивас, Ирван Белло, Макс Мороз,
Футан Пэн за отзыв.

Ссылки

Белло, И. Лямбдасети: Моделирование дальнодействующих взаимодействий без внимания. ICLR, 2021.

Белло, И., Федус, В., Ду, Х., Кубук, Э.Д., Шринивас, А., Лин, Т.-Й., Шленс, Дж. и Зоф, Б. Возвращаясь к сетям ResNet: улучшенные стратегии обучения и масштабирования. Препринт arXiv arXiv:2103.07579, 2021.

Бенжио, Ю., Лурадур, Ж., Коллобер, Р. и Уэстон, Дж. Обучение по учебной программе. ICML, 2009.

Брок, А., Де, С., Смит, С.Л. и Симонян, К. Высокопроизводительное распознавание крупномасштабных изображений без нормализации . Препринт arXiv arXiv:2102.06171, 2021.

Brown, TB, Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, DM, Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I. и Amodei, D. Языковые модели — это обучающиеся с небольшим количеством попыток. NeurIPS, 2020.

Cai, H., Zhu, L., and Han, S. Proxylessnas: Прямой поиск нейронной архитектуры на целевой задаче и оборудовании. ICLR, 2019.

Чэнь, И., Ян, Т., Чжан, С., Мэн, Г., Пан, К. и Сан, Дж. Детнас: Поиск нейронной архитектуры при обнаружении объектов. НейроИПС, 2019.

Cubuk, ED, Zoph, B., Shlens, J., и Le, QV Randaugment : Практическое автоматизированное дополнение данных с сокращенным пространством поиска. ECCV, 2020.

Dong, X., Tan, M., Yu, AW, Peng, D., Gabrys, B. и Le, QV Autohas: Эффективный поиск гиперпараметров и архитектуры. Препринт arXiv arXiv:2006.03656, 2020.

Досовицкий А., Бейер Л., Колесников А., Вайсенборн Д., Чжай К., Унтертинер Т., Дегани М., Миндерер М., Хейгольд Г., Гелли С., Ушкорейт Дж. и Хоулсби Н. Изображение стоит 16x16 слов: Трансформеры для распознавания изображений в масштабе. ICLR, 2021.

Элскен, Т., Метцен, Дж. Х. и Хаттер, Ф. Поиск нейронной архитектуры: обзор. Журнал исследований машинного обучения, 2019.

Гупта, С. и Акин, Б. Проектирование нейронных сетей с поддержкой ускорителей с использованием automl. Семинар по интеллектуальным возможностям на устройстве в SysML, 2020.

Гупта, С. и Тан, М. Efficientnet-edgetpu: Создание оптимизированных для ускорителей нейронных сетей с помощью auto-toml. https://ai.googleblog.com/2019/08/effectivenet-edgetpu-creating.html , 2019.

Хе, К., Чжан, С., Жэнь, С. и Сан, Дж. Глубокое остаточное обучение для распознавания изображений. CVPR, стр. 770–778, 2016.

Хоффер, Э., Вайнштейн, Б., Хубара, И., Бен-Нун, Т., Хёфлер, Т. и Соудри, Д. Смешивание и сопоставление: обучение сверточных сетей с изображениями разных размеров для повышения точности, скорости и устойчивости к масштабированию. Препринт arXiv arXiv:1908.08986, 2019.

Ховард, Дж. Обучение imagenet за 3 часа по 25 минут. https://www.fast.ai/2018/04/30/dawnbench-fastai/, 2018.

Хуан, Г., Сан, И., Лю, З., Седра, Д. и Вайнбергер, К. К. Глубокие сети со стохастической глубиной. ECCV, стр. 646–661, 2016.

Хуан, Г., Лю, З., Ван дер Маатен, Л. и Вайнбергер, К. К. Плотносвязанные сверточные сети. CVPR, 2017.

Хуан, И., Ченг, И., Чен, Д., Ли, Х., Нгиам, Дж., Ле, К. В. и Чен, З. Gpipe: Эффективное обучение гигантских нейронных сетей с использованием конвейерного параллелизма. NeurIPS, 2019.

Каррас, Т., Айла, Т., Лайне, С. и Лехтинен, Дж. Прогрессивное выращивание растений для улучшения качества, стабильности и вариативности. ICLR, 2018.

Колесников, А., Бейер, Л., Чжай, Х., Пучсервер, Дж., Юнг, Дж., Джелли, С. и Хоулсби, Н. Большой перенос (бит): общее обучение визуальному представлению. ECCV, 2020.

Краузе, Дж., Дэн, Дж., Старк, М. и Фей-Фей, Л. Сбор крупномасштабного набора данных по мелкозернистым автомобилям. Второй семинар по мелкозернистой визуальной категоризации, 2013.

Крижевский, А. и Хинтон, Г. Изучение нескольких слоев признаков на основе крошечных изображений. Технический отчет, 2009.

Ли, С., Тан, М., Панг, Р., Ли, А., Ченг, Л., Ле, К. и Джуппи, Н. Поиск быстрых семейств моделей для ускорителей центров обработки данных. CVPR, 2021.

Лю, Ч., Чэнь, Л.-Ч., Шрофф, Ф., Адам, Х., Хуа, В., Юйлле, А. и Фэй-Фэй, Л. Auto-deeplab: Иерархический поиск нейронной архитектуры для семантической сегментации изображений. ЦВПР, 2019.

Махаджан, Д., Гиршик, Р., Раманатан, В., Хе, К., Палури, М., Ли, Й., Бхарамбе, А. и ван дер Маатен, Л. Исследование пределов слабо контролируемой предварительной подготовки. Препринт arXiv arXiv:1805.00932, 2018.

Нильсбак, М.-Э. и Зиссерман, А. Автоматизированная классификация цветов по большому количеству классов. ICVGIP, стр. 722–729, 2008.

Пресс, О., Смит, Н.А. и Льюис, М. Shortformer: лучшее моделирование языка с использованием более коротких входных данных. Препринт arXiv arXiv:2012.15832, 2021.

Радосавович, И., Косарайу, Р.П., Гиршик, Р., Хе, К. и Доллар, П. Проектирование пространств сетевого проектирования. ЦВПР, 2020.

Ридник, Т., Лоуэн, Х., Ной, А., Барух, Э.Б., Шарир, Г. и Фридман, И. Треснет: Высокопроизводительная архитектура, ориентированная на графические процессоры. Препринт arXiv arXiv:2003.13630, 2020.

Русаковский, О., Дэн, Дж., Су, Х., Краузе, Дж., Сатиш, С., Ма, С., Хуан, З., Карпати, А., Хосла, А., Бернстайн, М. и др. Крупномасштабная задача визуального распознавания в Imagenet . Международный журнал компьютерного зрения, 115(3): 211–252, 2015.

Сандлер, М., Ховард, А., Чжу, М., Жмогинов, А. и Чен, Л.-К. Mobilenetv2: Обратные остатки и линейные узкие места. CVPR, 2018.

Сифре, Л. Рассеивание жестких движений для классификации изображений. Кандидатская диссертация, раздел 6.2, 2014.

Шринивас, А., Лин, Т.-Й., Пармар, Н., Шленс, Дж., Аббель, П. и Васвани, А. Трансформаторы «бутылочного горлышка» для визуального распознавания. Препринт arXiv arXiv:2101.11605, 2021.

Шривастава, Н., Хинтон, Г., Крижевский, А., Суцкевер, И. и Салахутдинов, Р. Dropout: простой способ предотвратить переобучение нейронных сетей. Журнал исследований машинного обучения, 15(1):1929–1958, 2014.

Тан, М. и Ле, QV Efficientnet: Переосмысление масштабирования модели для сверточных нейронных сетей. ICML, 2019a.

Тан, М. и Ле, К. В. Смешанная свертка по глубине. национальные ядра. BMVC, 2019b.

Тан, М., Чен, Б., Пан, Р., Васудеван, В. и Ле, К. В. Mnasnet: Платформенно-ориентированная нейронная архитектура поиска для мобильных устройств. CVPR, 2019.

Тан, М., Панг, Р. и Ле, QV Efficientdet: Масштабируемость и эффективное обнаружение объектов. CVPR, 2020.

Туврон, Х., Ведальди, А., Дузе, М. и Жегу, Х. Исправление расхождения в разрешении между поездом и тестом. Препринт arXiv arXiv :1906.06423, 2019.

Туврон, Х., Ведальди, А., Дузе, М. и Жегу, Х. Исправление расхождения в разрешении между поездом и тестом: Fixeffectivenet. Препринт arXiv arXiv:2003.08237, 2020.

Туврон, Х., Корд, М., Дузе, М., Масса, Ф., Саблейроль, А. и Жегу, Х. Обучение эффективных с точки зрения данных преобразователей изображений и дистилляция с помощью внимания. Препринт arXiv arXiv:2012.12877, 2021.

Уайтман, Р. Модель изображения Pytorch. https://github.com/rwightman/pytorch-image-models , Доступно 18 февраля 2021 г., 2021 г.

Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y. и Keutzer, K. Fbnet: Эффективное проектирование сверточных сетей с учетом аппаратного обеспечения посредством поиска дифференцируемой нейронной архитектуры. CVPR, 2019.

Се, К., Луонг, М.-Т., Хови, Э. и Ле, К.В. Самообучение с шумным учеником улучшает классификацию ImageNet . CVPR, 2020.

Xiong, Y., Liu, H., Gupta, S., Akin, B., Bender, G., Kindermans , P.-J., Tan, M., Singh, V. и Chen, B. Mobiledets: Поиск архитектур обнаружения объектов для мобильных ускорителей. Препринт arXiv arXiv:2004.14525, 2020.

Юй, Х., Лю, А., Лю, С., Ли, Г., Луо, П., Чэн, Р., Ян, Дж. и Чжан, К. Pda: Прогрессивное дополнение данных для общей надежности глубоких нейронных сетей. Препринт arXiv arXiv:1909.04839, 2019.

Юань, Л., Чэнь, И., Ван, Т., Ю, В., Ши, И., Тай, Ф. Э., Фэн, Дж. и Янь, С. Токены-в-токены vit: Обучение преобразователей зрения с нуля на imagenet. Препринт arXiv arXiv:2101.11986, 2021.

Чжан, Х., Сиссе, М., Дофин, Я. Н. и Лопес-Пас, Д. Mixup: за пределами эмпирической минимизации риска. ICLR, 2018.

Чжан, Х., У, Ч., Чжан, З., Чжу, И., Линь, Х., Чжан, З., Сан, И., Хе, Т., Мюллер, Дж., Манмата, Р., Ли, М. и Смола, А. Реснест: Сети с разделенным вниманием. Препринт arXiv arXiv:2012.12877, 2020.

Зоф, Б., Васудеван, В., Шленс, Дж. и Ле, К. В. Изучение переносимых архитектур для масштабируемого распознавания изображений. ЦВПР, 2018.