Designing Network Design Spaces

Ilija Radosavovic   Raj Prateek Kosaraju   Ross Girshick   Kaiming He   Piotr Dollár
Facebook AI Research (FAIR)

김성철

Contents

Previous research

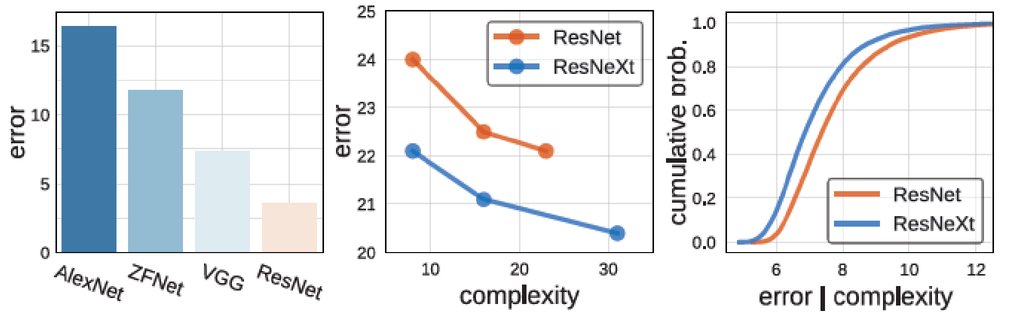
- Terminology
  - **Model family**
    - ResNet, DenseNet, …
  - **Design space**
    - A concrete set of architectures that can be instantiated from the model family
    - Two components
      - **A parametrization of a model family** such that
        specifying a set of model hyperparameters fully defines a network instantiation
      - **A set of allowable values** for each hyperparameter

Previous research

stage	operation	output
stem	3×3 conv	32×32×16
stage 1	{block}×d_1	32×32×w_1
stage 2	{block}×d_2	16×16×w_2
stage 3	{block}×d_3	8×8×w_3
head	pool + fc	1×1×10

	R-56	R-110
flops (B)	0.13	0.26
params (M)	0.86	1.73
error [8]	6.97	6.61
error [ours]	6.22	5.91

Table 1. **Design space parameterization**. *(Left)* The general network structure for the standard model families used in our work. Each stage consists of a sequence of d blocks with w output channels (block type varies between design spaces). *(Right)* Statistics of ResNet models [8] for reference. In our notation, R-56 has d_i=9 and w_i=8·2^i and R-110 doubles the blocks per stage d_i. We report original errors from [8] and those in our reproduction.



(a) point estimates (b) curve estimates (c) distribution estimates

Figure 1. **Comparing networks.** (a) Early work on neural networks for visual recognition tasks used *point estimates* to compare architectures, often *irrespective of model complexity*. (b) More recent work compares *curve estimates* of error *vs*. complexity traced by a handful of selected models. (c) We propose to *sample* models from a parameterized model design space, and measure *distribution estimates* to compare design spaces. This methodology allows for a more complete and unbiased view of the design landscape.

	depth	width	ratio	groups	total
`Vanilla`	1,24,9	16,256,12			1,259,712
`ResNet`	1,24,9	16,256,12			1,259,712
`ResNeXt-A`	1,16,5	16,256,5	1,4,3	1,4,3	11,390,625
`ResNeXt-B`	1,16,5	64,1024,5	1,4,3	1,16,5	52,734,375

Table 2. **Design spaces.** Independently for each of the three network stages i, we select the number of blocks d_i and the number of channels per block w_i. Notation a, b, n means we sample from n values spaced about evenly (in log space) in the range a to b. For the `ResNeXt` design spaces, we also select the bottleneck width ratio r_i and the number of groups g_i per stage. The total number of models is $(dw)^3$ and $(dwrg)^3$ for models w/o and with groups.

I. Radosavovic et al., On Network Design Spaces for Visual Recognition, ICCV 2019.

Introduction

- Manual network design
  - LeNet, AlexNet, VGG, ResNet, …
  - **Particular network instantiations** and **design principles** can be generalized and applied to numerous settings

- Finding well-optimized neworks
  - Neural Architecture Search (NAS)
  - High performance, but **NO** discovery of **network design principles**

Introduction

- **AnyNet**
  - Unconstrained design space

- **RegNet**
  - A low-dimensional design space consisting of simple "regular" networks
  - Stage width and depths are determined by a *quantized linear function*

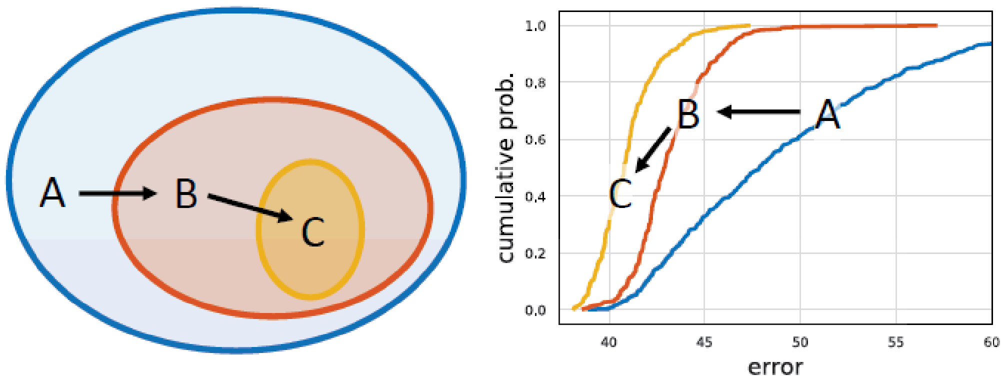
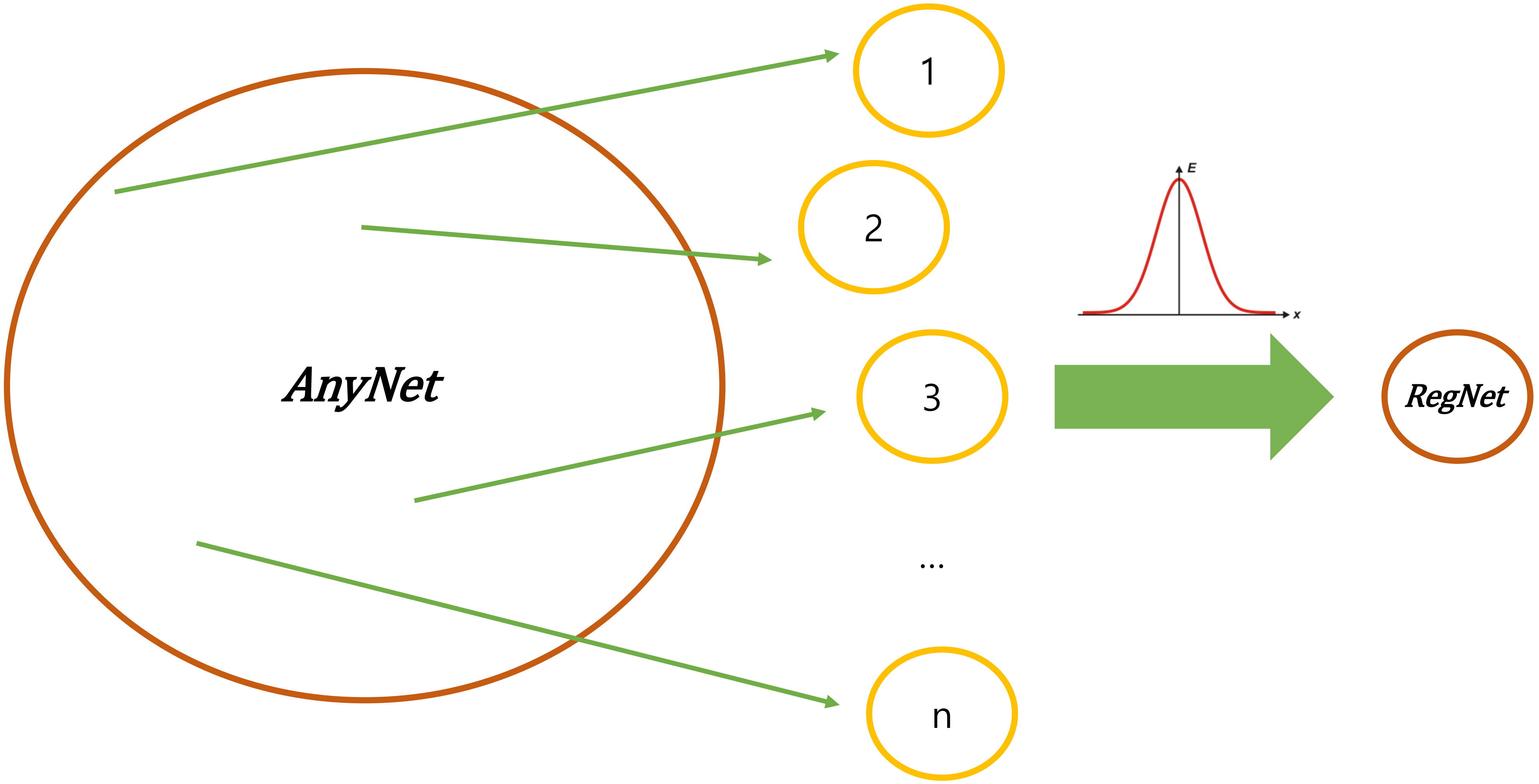


Figure 1. **Design space design.** We propose to *design* network design spaces, where a design space is a parametrized *set* of possible model architectures. Design space design is akin to manual network design, but elevated to the *population* level. In each step of our process the input is an initial design space and the output is a refined design space of simpler or better models. Following [21], we characterize the quality of a design space by sampling models and inspecting their *error distribution*. For example, in the figure above we start with an initial design space A and apply two refinement steps to yield design spaces B then C. In this case C ⊆ B ⊆A (left), and the error distributions are strictly improving from A to B to C (right). The hope is that *design principles* that apply to model populations are more likely to be robust and generalize.

Related work

- **Manual network design**
  - VGG, Inception, ResNet, ResNeXt, DenseNet, MobileNet, …
- **Automated network design**
  - NAS
- **Network scaling**
  - EfficientNet
- **Comparing networks**
- **Parameterization**
  - An empirical study justifying the design choices & insights into structural design choices

Design Space Design



Design Space Design

- **Tools for Design Space Design**
  - I. Radosavovic et al.
  - Quantify the quality of a design space by *sampling* a set of models from that design space and characterizing the resulting model error *distribution*
  - How to obtain a distribution of models?
    - **low-compute, low-epoch training regime!!!**
      - e.g. use the 400M flop regime and train each sampled model for 10 epochs

Design Space Design

- **Tools for Design Space Design**
  - The error *empirical distribution function* (EDF)

$$F(e)=\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}[e_i<e]. \qquad (1)$$

  - Compute and plot error EDFs to summarize design space quality
  - Visualize various properties of a design space & use an empirical bootstrap

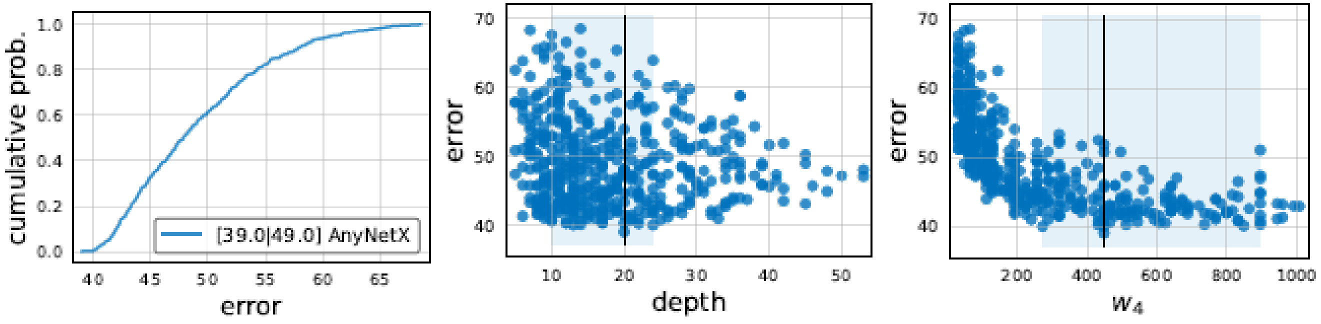


Figure 2. Statistics of the **AnyNetX design space** computed with $n = 500$ sampled models. *Left:* The error *empirical distribution function* (EDF) serves as our foundational tool for visualizing the quality of the design space. In the legend we report the min error and mean error (which corresponds to the area under the curve). *Middle:* Distribution of network depth d (number of blocks) versus error. *Right:* Distribution of block widths in the fourth stage (w_4) versus error. The blue shaded regions are ranges containing the best models with 95% confidence (obtained using an empirical bootstrap), and the black vertical line the most likely best value.

Design Space Design

- **The *AnyNet* Design Space**
  - The structure of the network
    → network depth, # of channels, bottleneck ratios, group widtl
  - **stem → body → head**
    - Stem and head are fixed and body is focused
  - X block : residual bottleneck + group conv
  - 16 degrees of freedom
    - 4 stages
    - # of blocks d_i, block width w_i, bottleneck ratio b_i, group width g_i
  - Log-uniform sampling
    - $d_i \leq 16, w_i \leq 1024$ and divisible by 8, $b_i \in \{1, 2, 4\}, g_i \in \{1, 2, \ldots, 32\}$

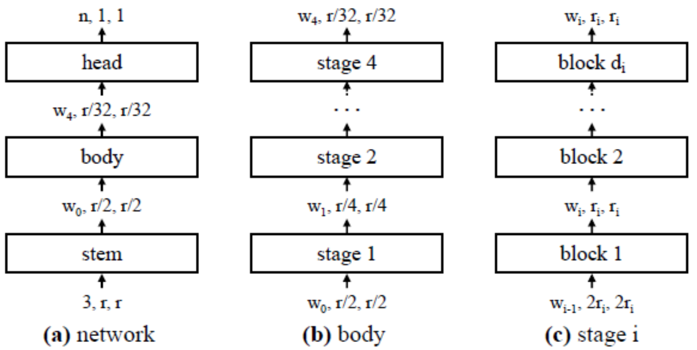


Figure 3. General **network structure** for models in our design spaces. (a) Each network consists of a stem (stride-two 3×3 conv with $w_0 = 32$ output channels), followed by the network body that performs the bulk of the computation, and then a head (average pooling followed by a fully connected layer) that predicts n output classes. (b) The network body is composed of a sequence of stages that operate at progressively reduced resolution r_i. (c) Each stage consists of a sequence of identical blocks, except the first block which uses stride-two conv. While the general structure is simple, the total number of possible network configurations is vast.

Design Space Design

- The *AnyNet* Design Space
  - $AnyNetX_A$: unconstrained $AnyNetX$ design space
  - $AnyNetX_B$: shared bottleneck ratio $b_i = b$ for all stages i for the $AnyNetX_A$ design space
  - $AnyNetX_C$: shared bottleneck ratio $g_i = g$ for all stages

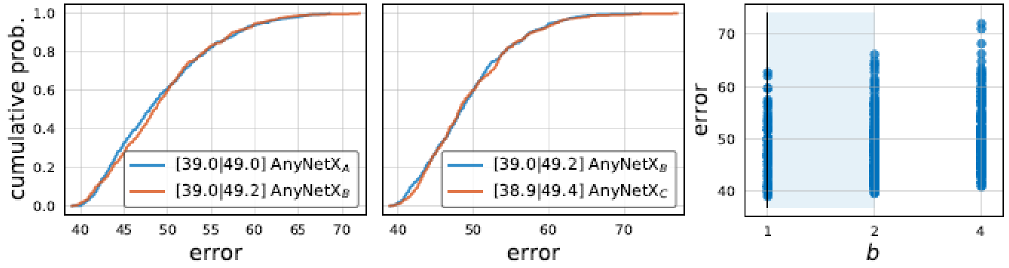


Figure 5. **$\texttt{AnyNetX}_\texttt{B}$** (left) and **$\texttt{AnyNetX}_\texttt{C}$** (middle) introduce a *shared* bottleneck ratio $b_i = b$ and shared group width $g_i = g$, respectively. This simplifies the design spaces while resulting in virtually no change in the error EDFs. Moreover, $\texttt{AnyNetX}_\texttt{B}$ and $\texttt{AnyNetX}_\texttt{C}$ are more amendable to analysis. Applying an empirical bootstrap to b and g we see trends emerge, *e.g.*, with 95% confidence $b \leq 2$ is best in this regime (right). No such trends are evident in the individual b_i and g_i in $\texttt{AnyNetX}_\texttt{A}$ (not shown).

Design Space Design

- The *AnyNet* Design Space
  - $AnyNetX_D : w_{i+1} \geq w_i$ from $AnyNetX_C$
  - $AnyNetX_E : d_{i+1} \geq d_i$ from $AnyNetX_D$
  - **The constraints on $\boldsymbol{w_i}$ and $\boldsymbol{d_i}$ each reduce the design space by 4!, with a cumulative reduction of $\boldsymbol{O(10^7)}$ from $\boldsymbol{AnyNetX_A}$**

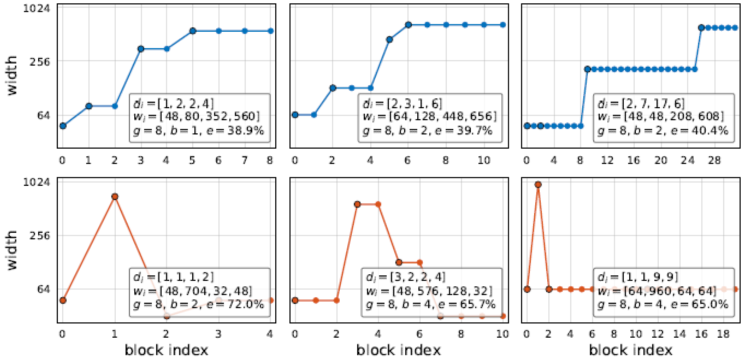


Figure 6. Example good and bad **AnyNetX$_C$ networks**, shown in the top and bottom rows, respectively. For each network, we plot the width w_j of every block j up to the network depth d. These *per-block* widths w_j are computed from the *per-stage* block depths d_i and block widths w_i (listed in the legends for reference).

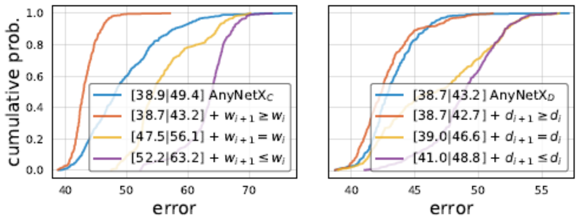


Figure 7. **AnyNetX$_D$** (left) and **AnyNetX$_E$** (right). We show various constraints on the per stage widths w_i and depths d_i. In both cases, having increasing w_i and d_i is beneficial, while using constant or decreasing values is much worse. Note that AnyNetX$_D$ = AnyNetX$_C$ + $w_{i+1} \geq w_i$, and AnyNetX$_E$ = AnyNetX$_D$ + $d_{i+1} \geq d_i$. We explore stronger constraints on w_i and d_i shortly.

Design Space Design

- The *RegNet* Design Space
  - (top-left) the best 20 models from $AnyNetX_E$
    - $w_j = 48 \cdot (j+1)$ for $0 \leq j \leq 20$
    - Trivial *linear fit* seems to explain the population trend!
  - A strategy to quantize a line
    - **A linear parameterization**

      $u_j = w_0 + w_a \cdot j$ for $0 \leq j < d$

      (depth d, initial width $w_0 > 0$, slope $w_a > 0$, block width u_j for each block $j < d$)

      compute s_j from $u_j = w_0 \cdot w_m^{s_j}$

      simply round s_j ($s_{\lfloor j \rceil}$) and compute quantized per-block width $w_j = w_0 \cdot w_m^{s_{\lfloor j \rceil}}$

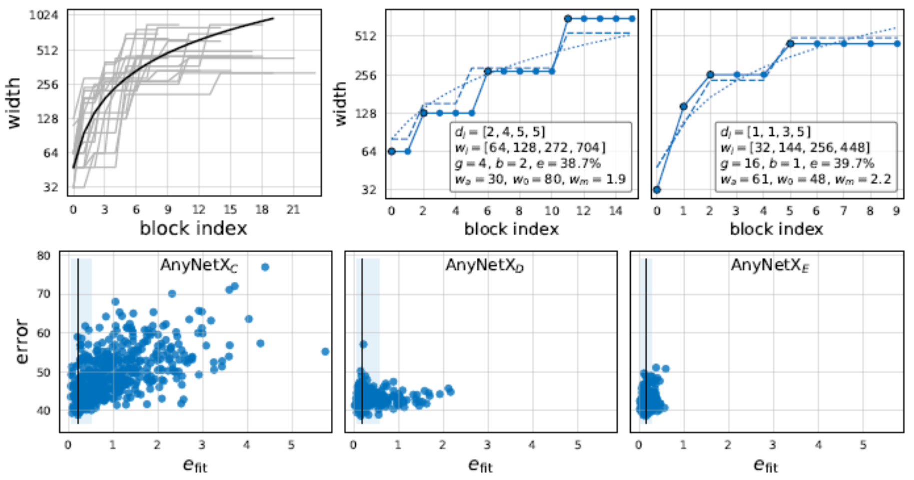


Figure 8. **Linear fits.** Top networks from the AnyNetX design space can be well modeled by a *quantized linear parameterization*, and conversely, networks for which this parameterization has a higher fitting error e_{fit} tend to perform poorly. See text for details.

Design Space Design

- The *RegNet* Design Space
  - Sample $d < 64, w_0, w_a < 256, 1.5 \le w_m \le 3$
  - (left) The error EDF of $RegNetX$
  - (middle) $w_m = 2$ (doubling width between stages) / $w_0 = w_a$ $(u_j = w_a \cdot (j+1))$
  - (right) random search efficiency

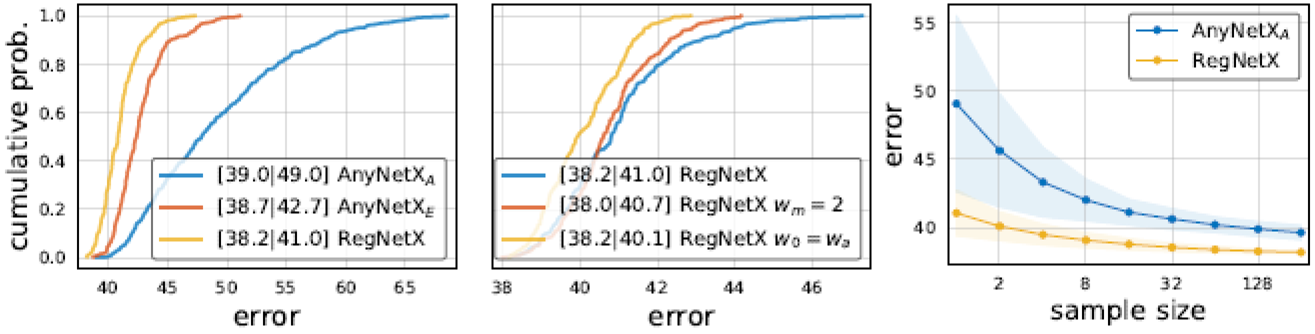


Figure 9. **RegNetX** design space. See text for details.

Design Space Design

- The *RegNet* Design Space
  - Sample $d < 64, w_0, w_a < 256, 1.5 \le w_m \le 3$
  - (left) The error EDF of *RegNetX*
  - (middle) $w_m = 2$ (doubling width between stages) / $w_0 = w_a$ $(u_j = w_a \cdot (j+1))$
  - (right) random search efficiency

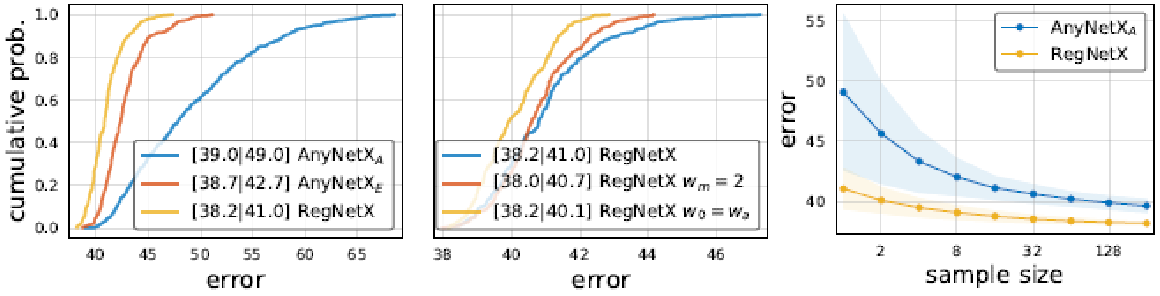


Figure 9. **RegNetX** design space. See text for details.

	restriction	dim.	combinations	total
AnyNetX$_A$	none	16	$(16{\cdot}128{\cdot}3{\cdot}6)^4$	$\sim 1.8{\cdot}10^{18}$
AnyNetX$_B$	$+\, b_{i+1} = b_i$	13	$(16{\cdot}128{\cdot}6)^4{\cdot}3$	$\sim 6.8{\cdot}10^{16}$
AnyNetX$_C$	$+\, g_{i+1} = g_i$	10	$(16{\cdot}128)^4{\cdot}3{\cdot}6$	$\sim 3.2{\cdot}10^{14}$
AnyNetX$_D$	$+\, w_{i+1} \ge w_i$	10	$(16{\cdot}128)^4{\cdot}3{\cdot}6/(4!)$	$\sim 1.3{\cdot}10^{13}$
AnyNetX$_E$	$+\, d_{i+1} \ge d_i$	10	$(16{\cdot}128)^4{\cdot}3{\cdot}6/(4!)^2$	$\sim 5.5{\cdot}10^{11}$
RegNet	quantized linear	6	$\sim 64^4{\cdot}6{\cdot}3$	$\sim 3.0{\cdot}10^8$

Table 1. **Design space summary.** See text for details.

Design Space Design

- **Design Space Generalization**
  - Discover general principles of network design that can *generalize* to new settings
  - Higher flops, higher epochs with 5-stage networks, with various block types

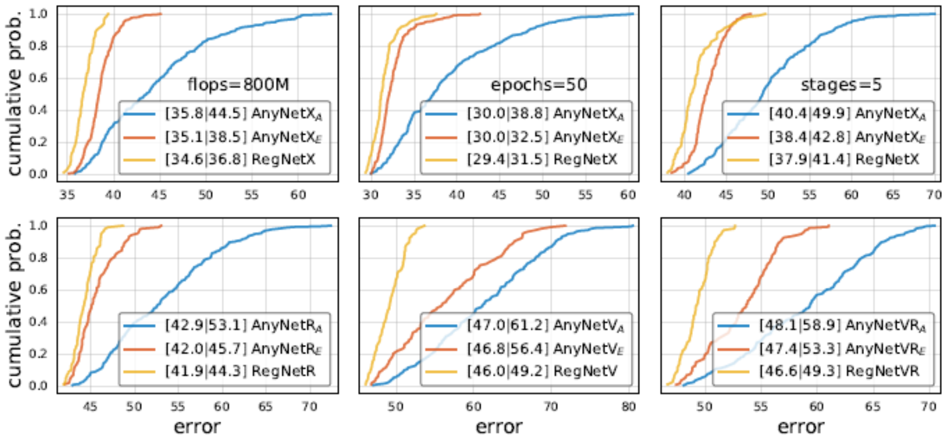


Figure 10. **RegNetX generalization.** We compare RegNetX to AnyNetX at higher flops (top-left), higher epochs (top-middle), with 5-stage networks (top-right), and with various block types (bottom). In all cases the ordering of the design spaces is consistent and we see no signs of design space overfitting.

Analyzing the *RegNetX* Design Space

- *RegNet* trends
  - (top-left) the depth d is stable across regimes

    → optimal depth : ~20 blocks (60 layers)
  - (top-middle) an optimal bottleneck ratio b is 1.0

    → effectively removes the bottleneck
  - (top-right) an optimal width multiplier w_m is ~2.5

    → similar but not identical to the popular recipe
  - (bottom) g, w_a, w_0 increase with complexity

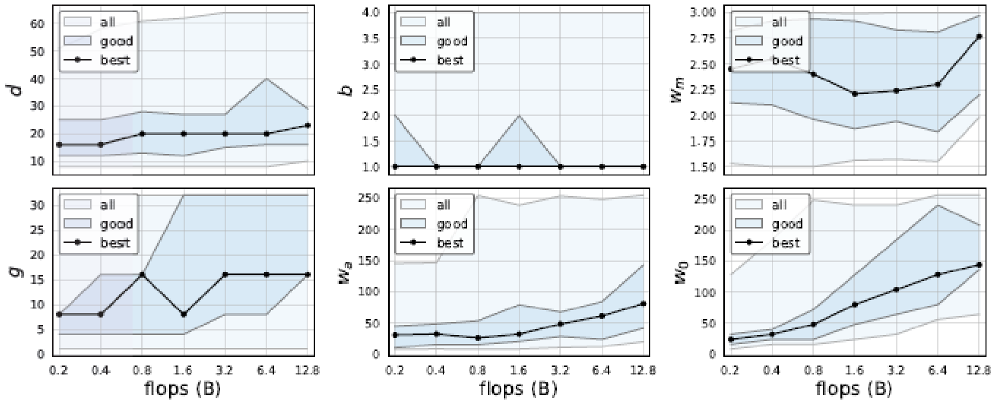


Figure 11. **RegNetX parameter trends**. For each parameter and each flop regime we apply an empirical bootstrap to obtain the range that contains best models with 95% confidence (shown with blue shading) and the likely best model (black line), see also Figure 2. We observe that for best models the depths d are remarkably stable across flops regimes, and $b = 1$ and $w_m \approx 2.5$ are best. Block and groups widths (w_a, w_0, g) tend to increase with flops.

Analyzing the *RegNetX* Design Space

- **Complexity analysis**
  - (top-left) *activation* : the size of the output tensors of all conv layers

    → activation can have a stronger correlation to runtime than flops
  - (bottom) complexity vs. flops

	flops	params	acts.
1×1 conv	w^2r^2	w^2	wr^2
3×3 conv	$3^2w^2r^2$	3^2w^2	wr^2
3×3 gr conv	3^2wgr^2	3^2wg	wr^2
3×3 dw conv	3^2wr^2	3^2w	wr^2

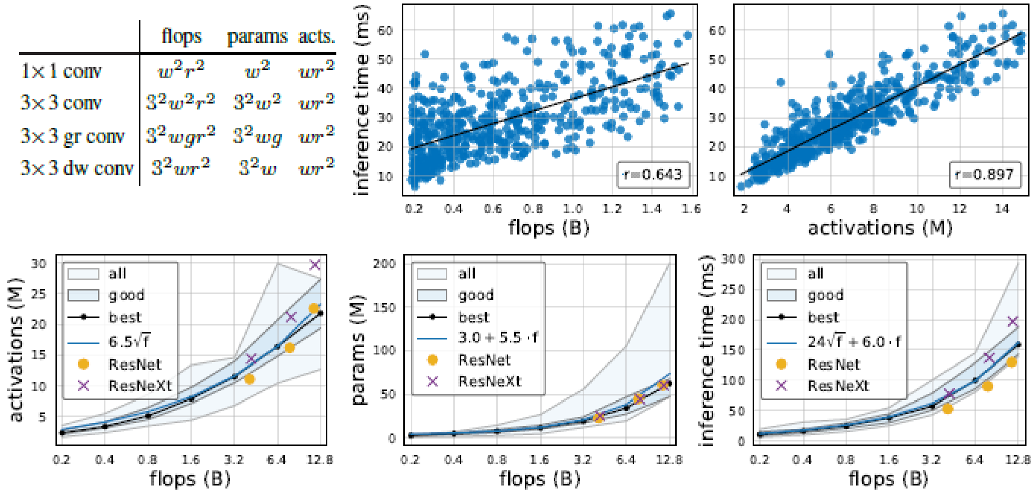


Figure 12. **Complexity metrics**. *Top:* Activations can have a stronger correlation to runtime on hardware accelerators than flops (we measure inference time for 64 images on an NVIDIA V100 GPU). *Bottom:* Trend analysis of complexity *vs.* flops and best fit curves (shown in blue) of the trends for best models (black curves).

Analyzing the *RegNetX* Design Space

- *RegNetX* constrained
  - $b = 1, d \leq 40, w_m \geq 2$ & limit parameters and activations

    → **fast, low-parameter, low-memory models without affecting accuracy**
  - The constrained version is superior across all flop regimes

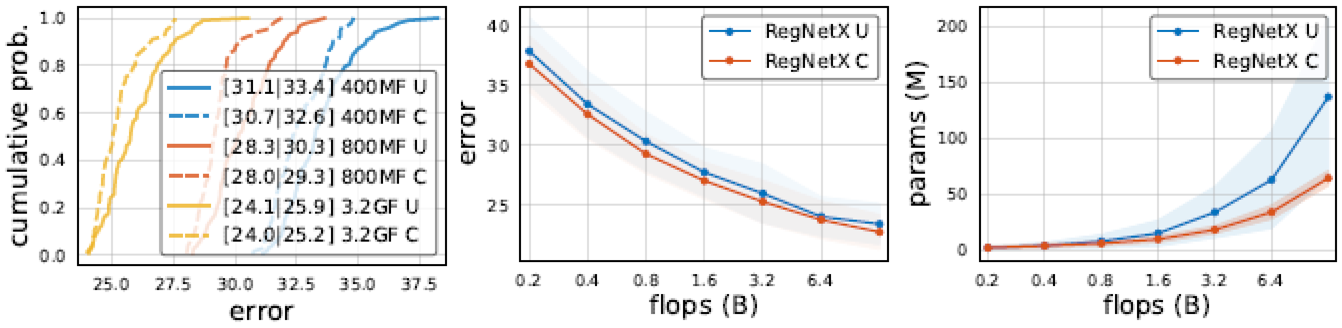


Figure 13. We **refine** **`RegNetX`** using various constraints (see text). The constrained variant (C) is best across all flop regimes while being more efficient in terms of parameters and activations.

Analyzing the *RegNetX* Design Space

- **Alternate design choices**
  - (left) the inverted bottleneck degrades the EDF and depthwise conv performs even worse
  - (middle) a fixed resolution of 224x224 is best
- **SE (Squeeze-and-Excitation)**
  - (right) X+SE=Y → *RegNetY* yields good gains

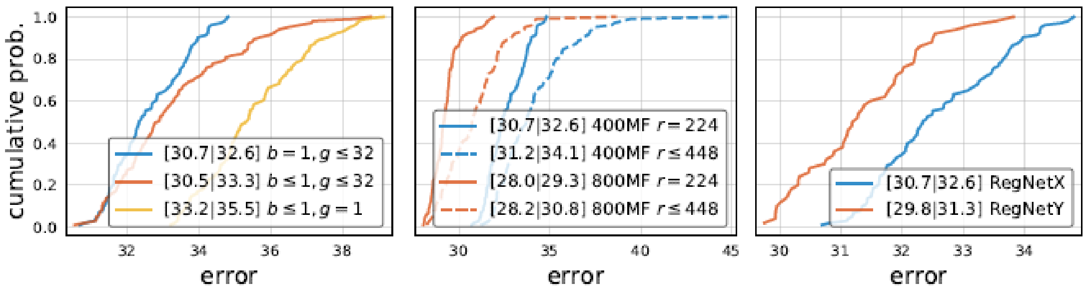


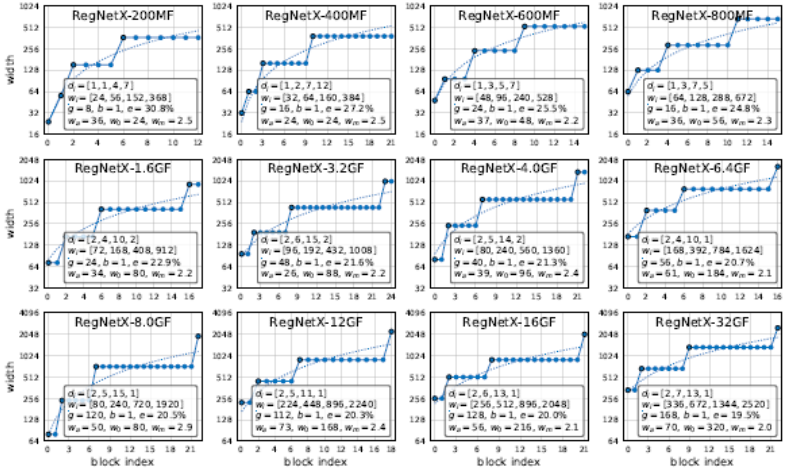
Figure 14. We evaluate RegNetX with **alternate design choices**. *Left:* Inverted bottleneck ($\frac{1}{8} \leq b \leq 1$) degrades results and depthwise conv ($g = 1$) is even worse. *Middle:* Varying resolution r harms results. *Right:* RegNetY (Y=X+SE) improves the EDF.

Comparison to Existing Networks

- $RegNetX$ and $RegNetY$ design spaces
  - The best model from 25 random settings of the $RegNet$ parameters (d, g, w_m, w_a, w_0)
  - Re-train the top model 5 times at 100 epochs
  - The higher flop models
    - a large number of blocks in the third stage and a small number of blocks in the last stage
  - The group width g increases with complexity, but depth d saturates for large models

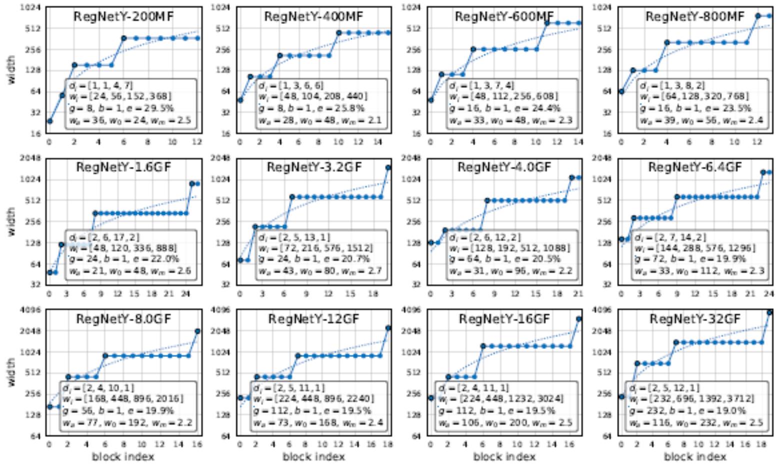
Comparison to Existing Networks

- *RegNetX* and *RegNetY* design spaces



	flops (B)	params (M)	acts (M)	batch size	infer (ms)	train (hr)	error (top-1)
REGNETX-200MF	0.2	2.7	2.2	1024	10	2.8	31.1±0.09
REGNETX-400MF	0.4	5.2	3.1	1024	15	3.9	27.3±0.15
REGNETX-600MF	0.6	6.2	4.0	1024	17	4.4	25.9±0.03
REGNETX-800MF	0.8	7.3	5.1	1024	21	5.7	24.8±0.09
REGNETX-1.6GF	1.6	9.2	7.9	1024	33	8.7	23.0±0.13
REGNETX-3.2GF	3.2	15.3	11.4	512	57	14.3	21.7±0.08
REGNETX-4.0GF	4.0	22.1	12.2	512	69	17.1	21.4±0.19
REGNETX-6.4GF	6.5	26.2	16.4	512	92	23.5	20.8±0.07
REGNETX-8.0GF	8.0	39.6	14.1	512	94	22.6	20.7±0.07
REGNETX-12GF	12.1	46.1	21.4	512	137	32.9	20.3±0.04
REGNETX-16GF	15.9	54.3	25.5	512	168	39.7	20.0±0.11
REGNETX-32GF	31.7	107.8	36.3	256	318	76.9	19.5±0.12

Figure 15. **Top REGNETX models.** We measure inference time for 64 images on an NVIDIA V100 GPU; train time is for 100 epochs on 8 GPUs with the batch size listed. Network diagram legends contain all information required to implement the models.



	flops (B)	params (M)	acts (M)	batch size	infer (ms)	train (hr)	error (top-1)
REGNETY-200MF	0.2	3.2	2.2	1024	11	3.1	29.6±0.11
REGNETY-400MF	0.4	4.3	3.9	1024	19	5.1	25.9±0.16
REGNETY-600MF	0.6	6.1	4.3	1024	19	5.2	24.5±0.07
REGNETY-800MF	0.8	6.3	5.2	1024	22	6.0	23.7±0.03
REGNETY-1.6GF	1.6	11.2	8.0	1024	39	10.1	22.0±0.08
REGNETY-3.2GF	3.2	19.4	11.3	512	67	16.5	21.0±0.05
REGNETY-4.0GF	4.0	20.6	12.3	512	68	16.8	20.6±0.08
REGNETY-6.4GF	6.4	30.6	16.4	512	104	26.1	20.1±0.04
REGNETY-8.0GF	8.0	39.2	18.0	512	113	28.1	20.1±0.09
REGNETY-12GF	12.1	51.8	21.4	512	150	36.0	19.7±0.06
REGNETY-16GF	15.9	83.6	23.0	512	189	45.6	19.6±0.16
REGNETY-32GF	32.3	145.0	30.3	256	319	76.0	19.0±0.12

Figure 16. **Top REGNETY models** (Y=X+SE). The benchmarking setup and the figure format is the same as in Figure 15.

Comparison to Existing Networks

- **State-of-the-Art Comparison: Mobile Regime**
  - The mobile regime : ~600MF
  - **NO enhancements for *RegNet***, while most mobile networks use various enhancements

	flops (B)	params (M)	top-1 error
MOBILENET [9]	0.57	4.2	29.4
MOBILENET-V2 [25]	0.59	6.9	25.3
SHUFFLENET [33]	0.52	-	26.3
SHUFFLENET-V2 [19]	0.59	-	25.1
NASNET-A [35]	0.56	5.3	26.0
AMOEBANET-C [23]	0.57	6.4	24.3
PNASNET-5 [17]	0.59	5.1	25.8
DARTS [18]	0.57	4.7	26.7
REGNETX-600MF	0.60	6.2	25.9$\pm$0.03
REGNETY-600MF	0.60	6.1	24.5$\pm$0.07

Table 2. **Mobile regime.** We compare existing models using *originally* reported errors to `RegNet` models trained in a *basic* setup. Our *simple* `RegNet` models achieve surprisingly good results given the effort focused on this regime in the past few years.

Comparison to Existing Networks

- **Standard Baselines Comparison: ResNe(X)t**

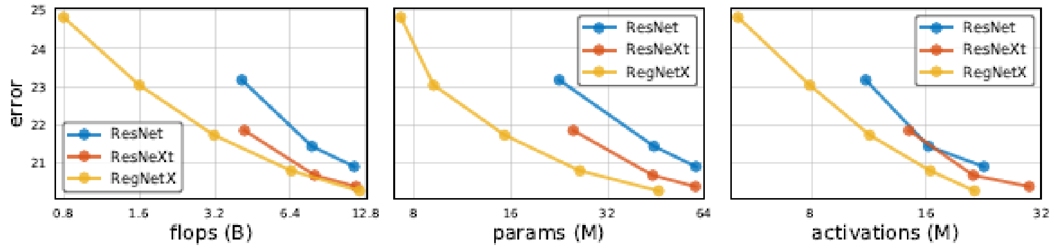


Figure 17. **ResNe(X)t comparisons.** REGNETX models versus RESNE(X)T-(50,101,152) under various complexity metrics. As all models use the identical components and training settings, all observed gains are from the design of the RegNetX design space.

	flops (B)	params (M)	acts (M)	infer (ms)	train (hr)	top-1 error ours±std [orig]
RESNET-50	4.1	22.6	11.1	53	12.2	23.2±0.09 [23.9]
REGNETX-3.2GF	3.2	15.3	11.4	57	14.3	**21.7**±0.08
RESNEXT-50	4.2	25.0	14.4	78	18.0	21.9±0.10 [22.2]
RESNET-101	7.8	44.6	16.2	90	20.4	21.4±0.11 [22.0]
REGNETX-6.4GF	6.5	26.2	16.4	92	23.5	**20.8**±0.07
RESNEXT-101	8.0	44.2	21.2	137	31.8	20.7±0.08 [21.2]
RESNET-152	11.5	60.2	22.6	130	29.2	20.9±0.12 [21.6]
REGNETX-12GF	12.1	46.1	21.4	137	32.9	**20.3**±0.04

(a) Comparisons grouped by **activations**.

	flops (B)	params (M)	acts (M)	infer (ms)	train (hr)	top-1 error ours±std [orig]
RESNET-50	4.1	22.6	11.1	53	12.2	23.2±0.09 [23.9]
RESNEXT-50	4.2	25.0	14.4	78	18.0	21.9±0.10 [22.2]
REGNETX-4.0GF	4.0	22.1	12.2	69	17.1	**21.4**±0.19
RESNET-101	7.8	44.6	16.2	90	20.4	21.4±0.11 [22.0]
RESNEXT-101	8.0	44.2	21.2	137	31.8	**20.7**±0.08 [21.2]
REGNETX-8.0GF	8.0	39.6	14.1	94	22.6	**20.7**±0.07
RESNET-152	11.5	60.2	22.6	130	29.2	20.9±0.12 [21.6]
RESNEXT-152	11.7	60.0	29.7	197	45.7	20.4±0.06 [21.1]
REGNETX-12GF	12.1	46.1	21.4	137	32.9	**20.3**±0.04

(b) Comparisons grouped by **flops**.

Table 3. **RESNE(X)T comparisons.** (a) Grouped by activations, REGNETX show considerable gains (note that for each group GPU inference and training times are similar). (b) REGNETX models outperform RESNE(X)T models under fixed flops as well.

Comparison to Existing Networks

- State-of-the-Art Comparison: Full Regime

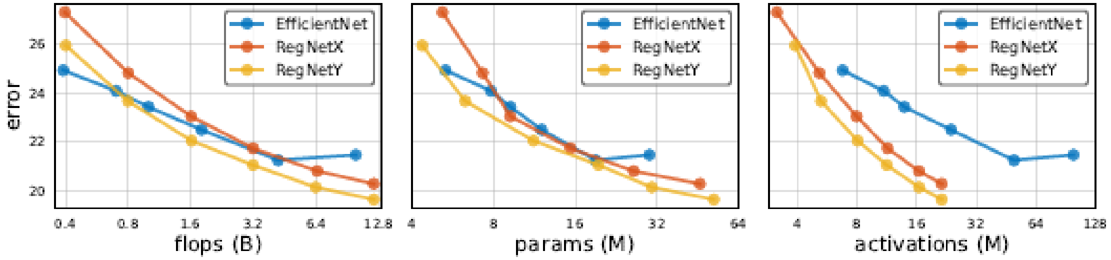


Figure 18. **EFFICIENTNET comparisons.** REGNETs outperform the state of the art, especially when considering activations.

	flops (B)	params (M)	acts (M)	batch size	infer (ms)	train (hr)	top-1 error ours±std [orig]
EFFICIENTNET-B0	0.4	5.3	6.7	256	34	11.7	**24.9**±0.03 [23.7]
REGNETY-400MF	0.4	4.3	**3.9**	1024	19	5.1	25.9±0.16
EFFICIENTNET-B1	0.7	7.8	10.9	256	52	15.6	**24.1**±0.16 [21.2]
REGNETY-600MF	0.6	6.1	**4.3**	1024	19	5.2	24.5±0.07
EFFICIENTNET-B2	1.0	9.2	13.8	256	68	18.4	**23.4**±0.06 [20.2]
REGNETY-800MF	0.8	6.3	**5.2**	1024	22	6.0	23.7±0.03
EFFICIENTNET-B3	1.8	12.0	23.8	256	114	32.1	22.5±0.05 [18.9]
REGNETY-1.6GF	1.6	11.2	**8.0**	1024	39	10.1	**22.0**±0.08
EFFICIENTNET-B4	4.2	19.0	48.5	128	240	65.1	21.2±0.06 [17.4]
REGNETY-4.0GF	4.0	20.6	**12.3**	512	68	16.8	**20.6**±0.08
EFFICIENTNET-B5	9.9	30.0	98.9	64	504	135.1	21.5±0.11 [16.7]
REGNETY-8.0GF	8.0	39.2	**18.0**	512	113	28.1	**20.1**±0.09

Table 4. **EFFICIENTNET comparisons** using our standard training schedule. Under *comparable training settings*, REGNETY outperforms EFFICIENTNET for most flop regimes. Moreover, REGNET models are considerably faster, *e.g.*, REGNETX-F8000 is about 5× *faster* than EFFICIENTNET-B5. Note that originally reported errors for EFFICIENTNET (shown grayed out), are much lower but use longer and enhanced training schedules, see Table 7.

감 사 합 니 다