PROGRESSIVE GROWING OF GANS FOR IMPROVED QUALITY, STABILITY, AND VARIATION

2018 ICLR oral

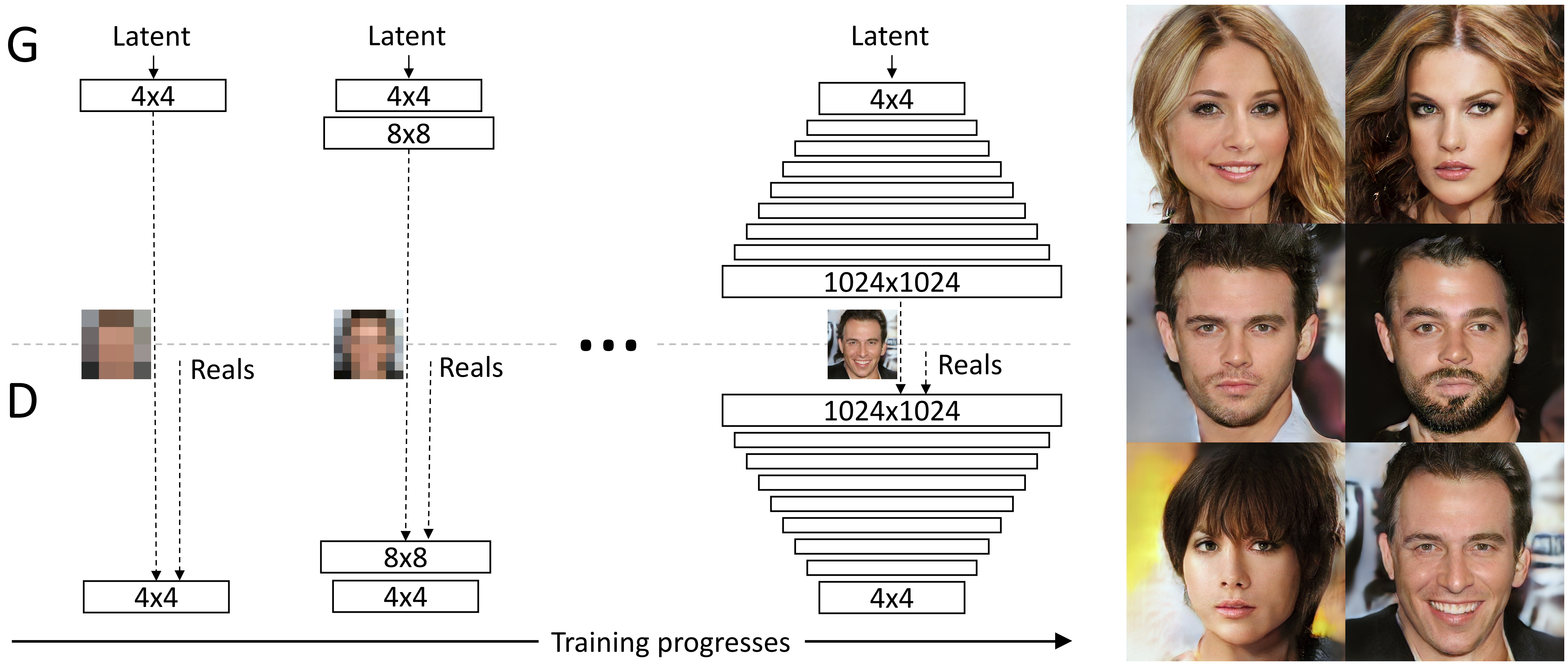
Wenjing Wang 20180506

1024 × 1024 images generated using the CELEBA-HQ dataset.

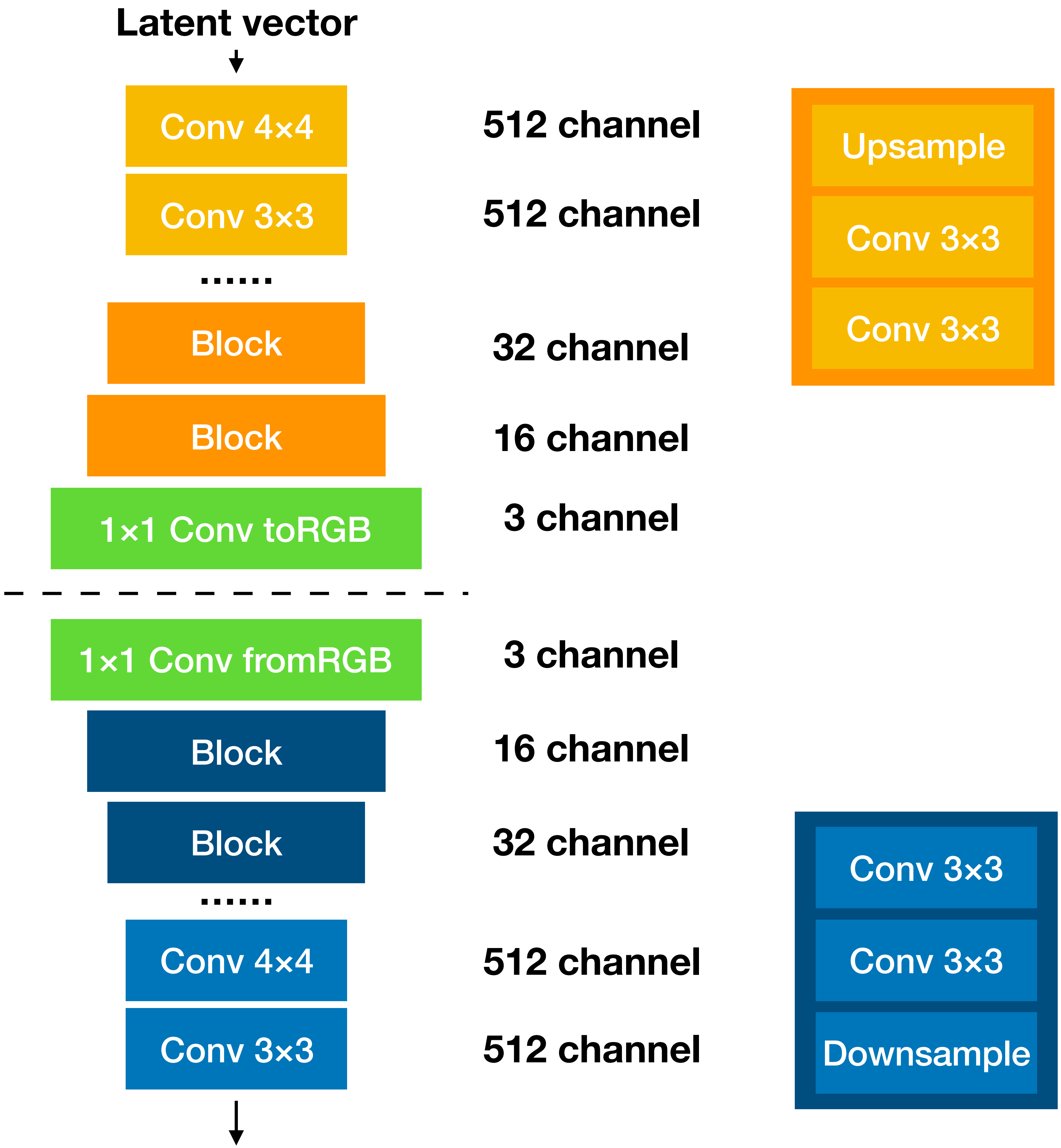
Overview

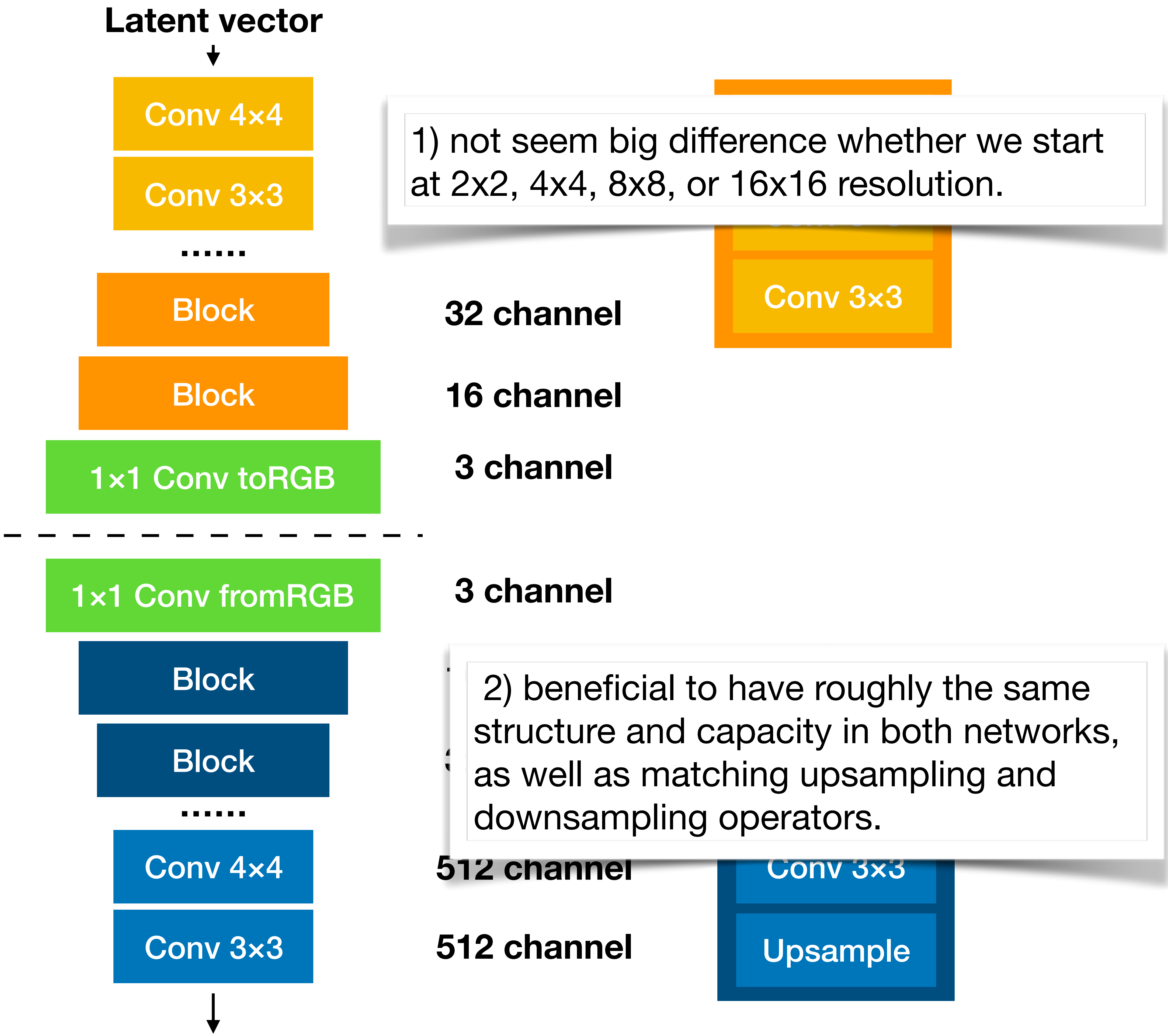
- **A new training methodology for GAN:**
  - grow both the generator and discriminator progressively
  - starting from a low resolution, add new layers that model increasingly fine details as training progresses.
- **A simple way to increase the variation in generated images**
- **Two implementation details that are important for discouraging unhealthy competition between the generator and discriminator.**
- **A new metric for evaluating GAN results, both in terms of image quality and variation.**

PROGRESSIVE GROWING OF GANS



- As the training advances, incrementally add layers to G and D, increasing the spatial resolution of the generated images.
- All existing layers remain trainable throughout the process.


Latent vector
Conv 4×4
512 channel
Conv 3×3
512 channel
......
Block
32 channel
Block
16 channel
1×1 Conv toRGB
3 channel
Upsample
Conv 3×3
Conv 3×3
1×1 Conv fromRGB
3 channel
Block
16 channel
Block
32 channel
......
Conv 4×4
512 channel
Conv 3×3
512 channel
Conv 3×3
Conv 3×3
Downsample

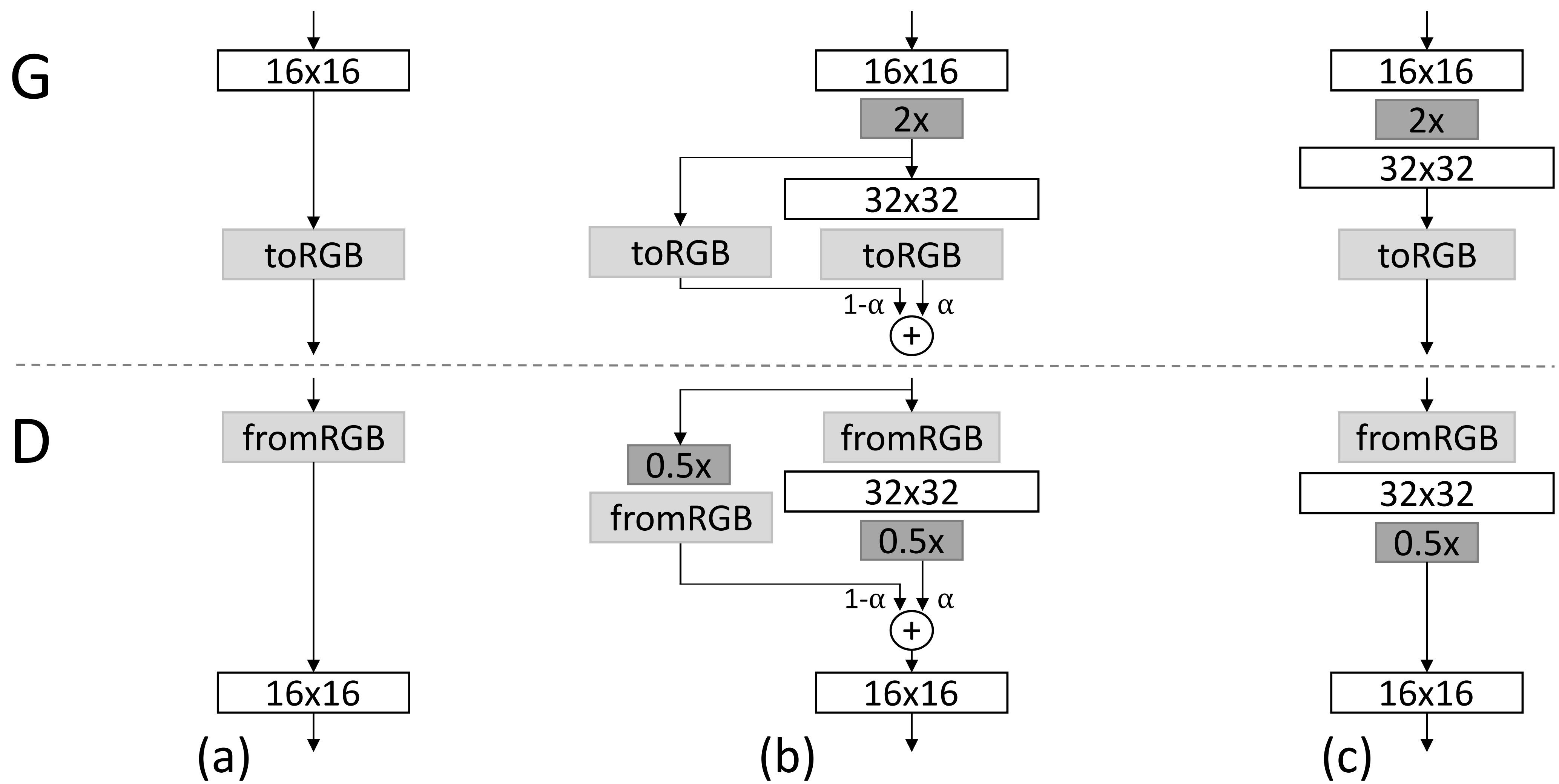


1) not seem big difference whether we start at 2x2, 4x4, 8x8, or 16x16 resolution.

2) beneficial to have roughly the same structure and capacity in both networks, as well as matching upsampling and downsampling operators.

PROGRESSIVE GROWING OF GANS

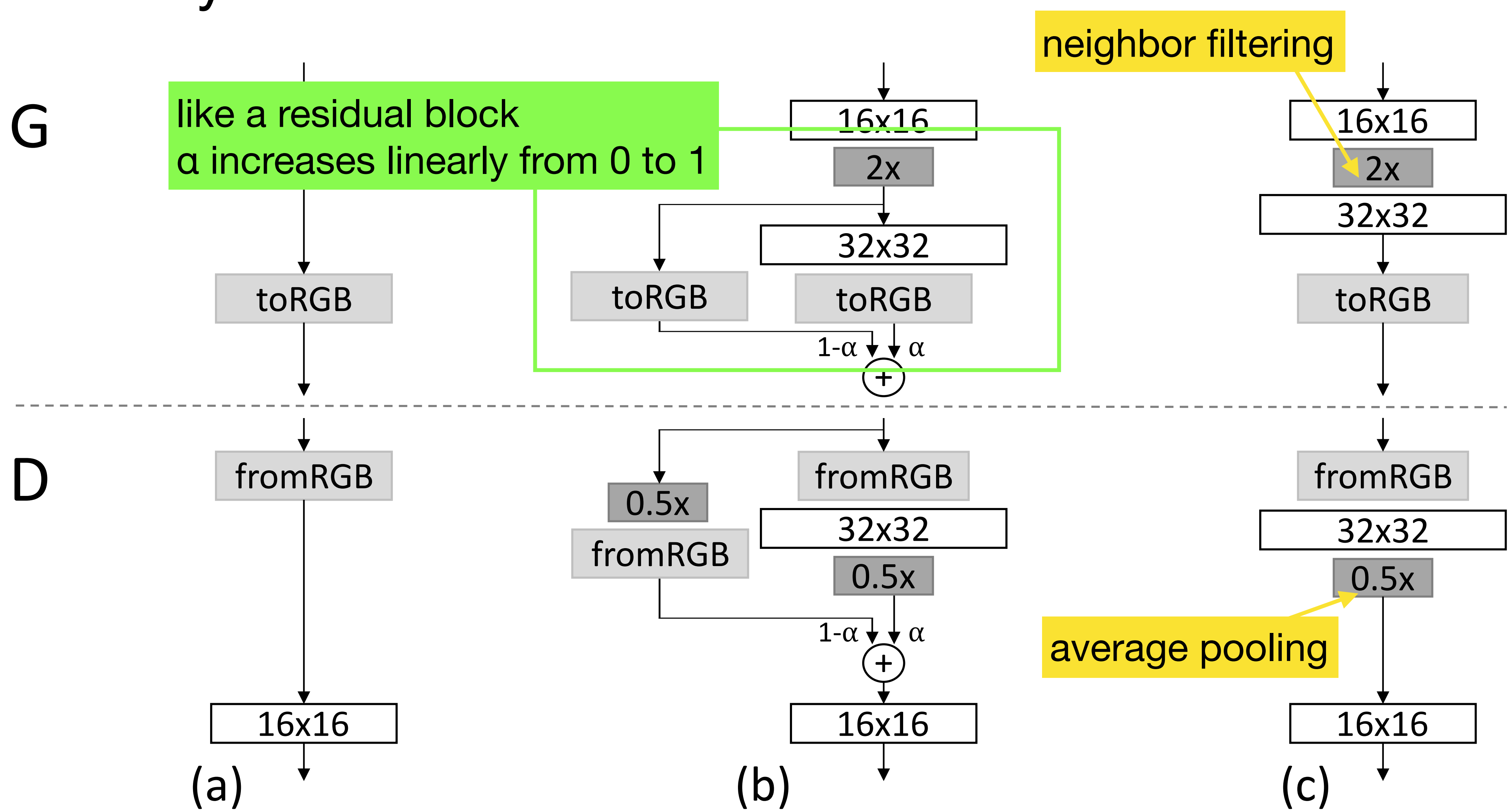
- When new layers are added to the networks, fade them in smoothly



transition (b) from 16 × 16 images (a) to 32 × 32 images (c)

PROGRESSIVE GROWING OF GANS

- When new layers are added to the networks, fade them in smoothly



transition (b) from 16 × 16 images (a) to 32 × 32 images (c)

Training strategy:

4×4 resolution, train discriminator until 800k real images.

Then alternate between:

1, fade in the 3-layer block during the next 800k images

2, stabilize the networks for 800k images

Minibatch size:

16 for resolutions 4^2–128^2 and then gradually decrease the size according to $256^2 \rightarrow 14$, $512^2 \rightarrow 6$, and $1024^2 \rightarrow 3$.

Loss:

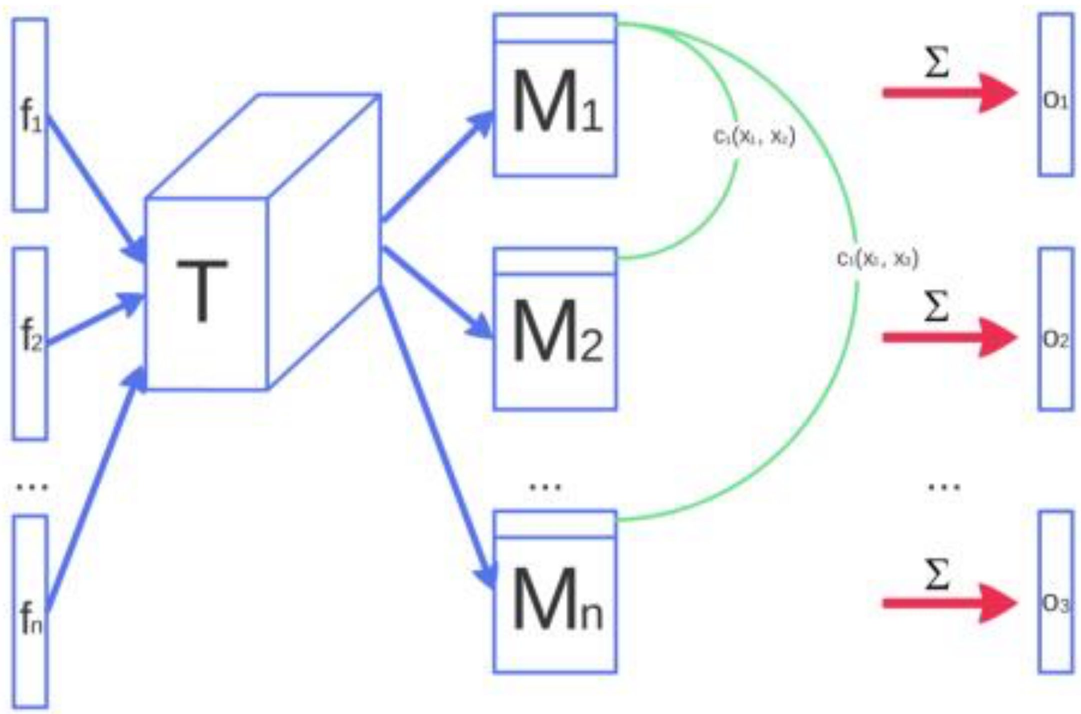
a variation of WGAN-GP loss. (LSGAN is generally a less stable loss function than WGAN-GP, and also has a tendency to lose some of the variation towards the end of long runs)

PROGRESSIVE GROWING OF GANS

- **For generator:**
- When new layers are added to the networks, fade them in smoothly
- **For discriminator:**
    - images downscaled to match the current resolution.
    - During a resolution transition, interpolate between two resolutions of the real images.

INCREASING VARIATION

- Salimans et al. (2016) suggest "minibatch discrimination"
  - adding a minibatch layer towards the end of the discriminator (compute feature statistics not only from individual images but also across the minibatch)



- We simplify this approach drastically while also improving the variation.

INCREASING VARIATION

- Compute the standard deviation for each feature in each spatial location over the minibatch
- Average these estimates over all features and spatial locations to arrive at a single value.
- Concatenate the single value to all spatial locations and over the minibatch.

- This layer could be inserted anywhere in the discriminator, but we have found it best to insert it towards the end.

NORMALIZATION IN G AND D

- **Problem: Unhealthy competition between G and D**
  - **Most other works**: using a variant of batch normalization in the generator, and often also in the discriminator.
    - There are for eliminate covariate shift 消除协变
  - **We** believe that the actual need in GANs is:
    - constraining signal magnitudes and competition

NORMALIZATION IN G AND D

- **Equalized Learning Rate**
- We use a N (0, 1) initialization and then scale the weights at runtime.
  - $w^i = w^i/c$, w^i: weights, c: per-layer normalization constant from He's initializer
- Benefit: relates to the scale-invariance in commonly used adaptive stochastic gradient descent methods (such as RMSProp and Adam)
- Our approach ensures that the dynamic range, and thus the learning speed, is the same for all weights.

NORMALIZATION IN G AND D

- **Equalized Learning Rate**
- We use a N (0
at runtime.
  - $w^i = w^i/c$, w
constant fr
- Benefit: relate
adaptive stoc
RMSProp and
- Our approach
the learning s

These sgd methods normalize a gradient update by its estimated standard deviation,

→ the update independent of the scale of the parameter

→ if some parameters have a larger dynamic range than others, they will take longer to adjust

→ a learning rate is both too large and too small at the same time.

NORMALIZATION IN G AND D

- **Equalized Learning Rate**
- We use a N (0, 1) initialization and then scale the weights at runtime.
  - $w^i = w^i/c$, w^i: weights, c: per-layer normalization constant from He's initializer
- Benefit: relates to the scale-invariance in commonly used adaptive stochastic gradient descent methods (such as RMSProp and Adam)
- Our approach ensures that the dynamic range, and thus the learning speed, is the same for all weights.

NORMALIZATION IN G AND D

- **Pixelwise Feature Vector Normalization In Generator**
- We normalize the feature vector in each pixel to unit length in the generator after each convolutional layer. We do this using a variant of “local response normalization”

$$b_{x,y} = a_{x,y} / \sqrt{\frac{1}{N} \sum_{j=0}^{N-1} (a^j_{x,y})^2 + \epsilon}, \text{ where } \epsilon = 10^{-8},$$

- N is the number of feature maps
- We find it surprising that this heavy-handed constraint does not seem to harm the generator in any way, and indeed with most datasets it does not change the results much, but it prevents the escalation of signal magnitudes very effectively when needed.

Don’t know why but it’s useful

NEW METRIC FOR EVALUATING GAN

- Existing methods such as MS-SSIM (Odena et al., 2017) find large-scale mode collapses reliably, but:
  - fail to react to smaller effects (*e.g.* loss of variation in colors or textures)
  - do not directly assess image quality in terms of similarity to the training set.

NEW METRIC FOR EVALUATING GAN

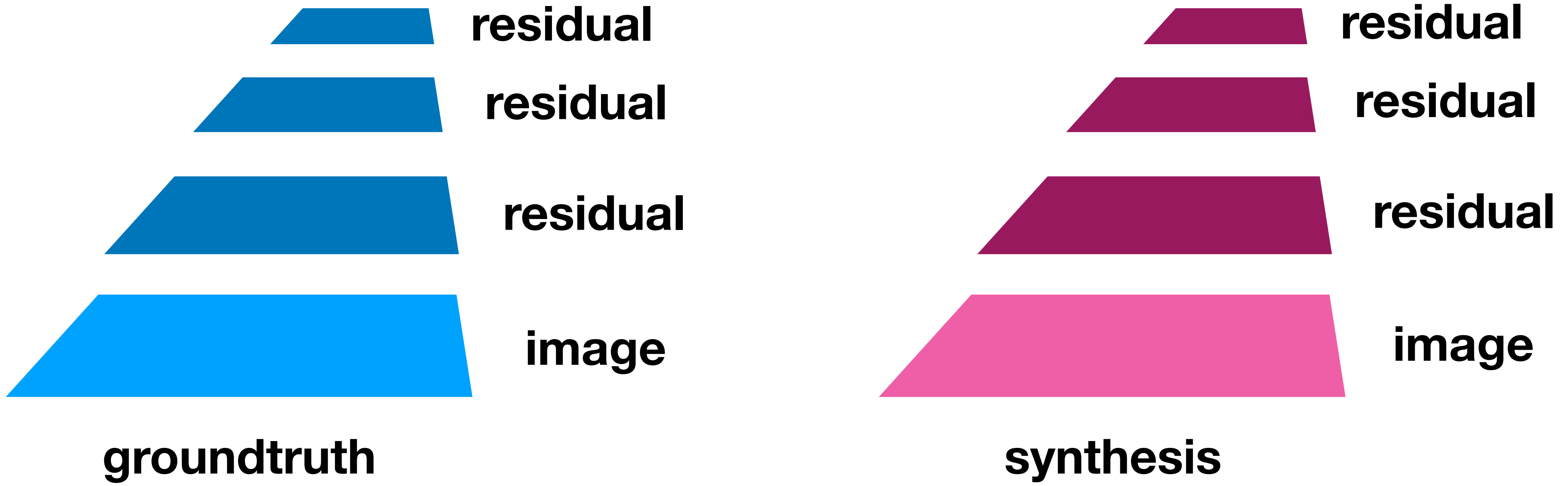
- We think: local image structure should be similar to the training set over all scales.
- We calculate: the multi-scale statistical similarity between distributions of local image patches drawn from Laplacian pyramid representations of generated and target images, starting at a low-pass resolution of 16×16 pixels.

NEW METRIC FOR EVALUATING GAN

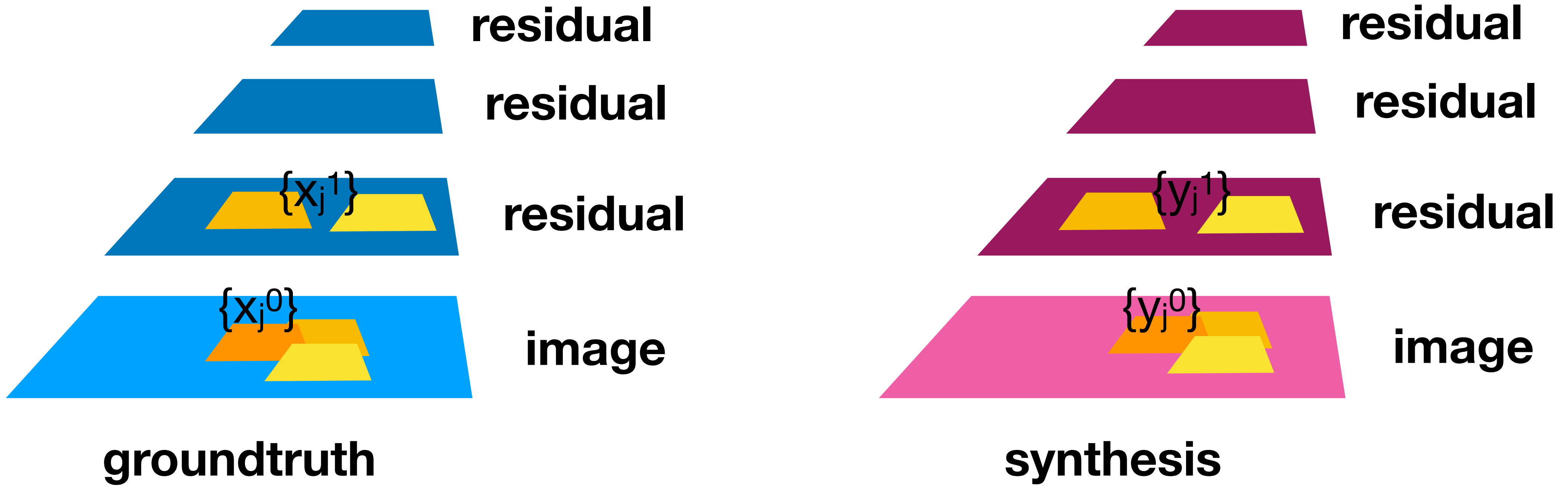
- We think: local image structure should be similar to the training set over all scales.
- We calculate: the multi-scale statistical similarity between distributions of local image patches drawn from Laplacian pyramid representations of generated and target images, starting at a low-pass resolution of 16×16 pixels.

NEW METRIC FOR EVALUATING GAN

- Laplacian pyramid

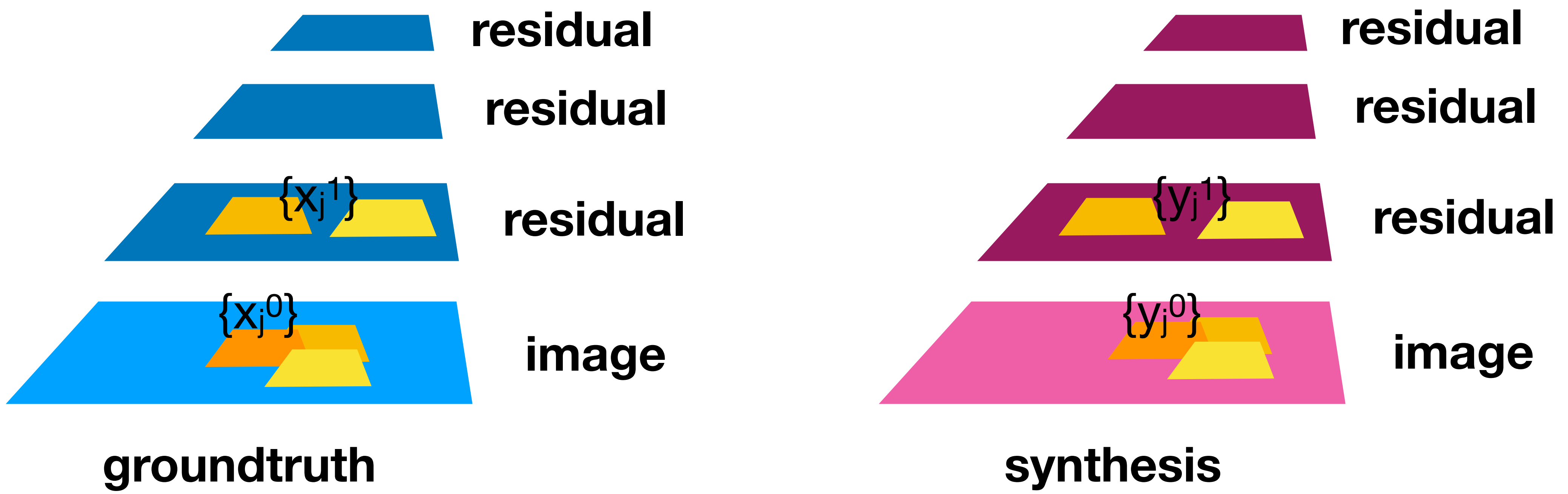


NEW METRIC FOR EVALUATING GAN



- randomly sample 16384 images and extract 128 descriptors from each level in the Laplacian pyramid
  - descriptor： 7 ×7 pixel neighborhood with 3 color channels
- Normalize $\{x_j\}$ and $\{y_j\}$ and calculate sliced Wasserstein distance SWD

NEW METRIC FOR EVALUATING GAN



- a small Wasserstein distance → the distribution of the patches is similar → the training images and generator samples appear similar in both appearance and variation at this spatial resolution.
- lowest-level patches → similarity in large-scale image structures; finest-level patches → pixel-level attributes

Experiment

Dataset

- CelebA
  - Large-scale CelebFaces Attributes (CelebA) Dataset
  - 202,599 images and 10,177 subjects. 5 landmark locations, 40 binary attributes.
- LSUN BEDROOM
  - LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop
  - one million labeled images for each of 10 scene categories and 20 object categories

Ablation Study

Training configuration	CELEBA						LSUN BEDROOM					
	Sliced Wasserstein distance $\times 10^3$					MS-SSIM	Sliced Wasserstein distance $\times 10^3$					MS-SSIM
	128	64	32	16	Avg		128	64	32	16	Avg	
(a) Gulrajani et al. (2017)	12.99	7.79	7.62	8.73	9.28	0.2854	11.97	10.51	8.03	14.48	11.25	**0.0587**
(b) + Progressive growing	4.62	**2.64**	3.78	6.06	4.28	**0.2838**	7.09	6.27	7.40	9.64	7.60	0.0615
(c) + Small minibatch	75.42	41.33	41.62	26.57	46.23	0.4065	72.73	40.16	42.75	42.46	49.52	0.1061
(d) + Revised training parameters	9.20	6.53	4.71	11.84	8.07	0.3027	7.39	5.51	3.65	9.63	6.54	0.0662
(e*) + Minibatch discrimination	10.76	6.28	6.04	16.29	9.84	0.3057	10.29	6.22	5.32	11.88	8.43	0.0648
(e) Minibatch stddev	13.94	5.67	2.82	5.71	7.04	0.2950	7.77	5.23	3.27	9.64	6.48	0.0671
(f) + Equalized learning rate	4.42	3.28	2.32	7.52	4.39	0.2902	**3.61**	3.32	**2.71**	6.44	4.02	0.0668
(g) + Pixelwise normalization	**4.06**	3.04	**2.02**	**5.13**	**3.56**	0.2845	3.89	**3.05**	3.24	**5.87**	**4.01**	0.0640
(h) Converged	2.42	2.17	2.24	4.99	2.96	0.2828	3.47	2.60	2.30	4.87	3.31	0.0636

- low-capacity network structure to amplify the differences between training configurations
- terminating the training once the discriminator has been shown a total of 10M real images -> the results are not fully converged

SWD and MSSSIM

Improved training of Wasserstein GANs

Training configuration	CELEBA						LSUN BEDROOM					
	Sliced Wasserstein distance $\times 10^3$					MS-SSIM	Sliced Wasserstein distance $\times 10^3$					MS-SSIM
	128	64	32	16	Avg		128	64	32	16	Avg	
(a) Gulrajani et al. (2017)	12.99	7.79	7.62	8.73	9.28	0.2854	11.97	10.51	8.03	14.48	11.25	**0.0587**
(b) + Progressive growing	4.62	**2.64**	3.78	6.06	4.28	**0.2838**	7.09	6.27	7.40	9.64	7.60	0.0615
(c) + Small minibatch	75.42	41.33	41.62	26.57	46.23	0.4065	72.73	40.16	42.75	42.46	49.52	0.1061
(d) + Revised training parameters	9.20	6.53	4.71	11.84	8.07	0.3027	7.39	5.51	3.65	9.63	6.54	0.0662
(e*) + Minibatch discrimination	10.76	6.28	6.04	16.29	9.84	0.3057	10.29	6.22	5.32	11.88	8.43	0.0648
(e) Minibatch stddev	13.94	5.67	2.82	5.71	7.04	0.2950	7.77	5.23	3.27	9.64	6.48	0.0671
(f) + Equalized learning rate	4.42	3.28	2.32	7.52	4.39	0.2902	**3.61**	3.32	**2.71**	6.44	4.02	0.0668
(g) + Pixelwise normalization	**4.06**	3.04	**2.02**	**5.13**	**3.56**	0.2845	3.89	**3.05**	3.24	**5.87**	**4.01**	0.0640
(h) Converged	2.42	2.17	2.24	4.99	2.96	0.2828	3.47	2.60	2.30	4.87	3.31	0.0636

(a) (b) (c) (d) (e∗) (e) (f) (g) (h) Converged

Notice that some images show aliasing and some are not sharp – this is a flaw of the dataset, which the model learns to replicate faithfully.

Our CELEBA-HQ results

Nearest neighbors found from the training data, based on feature-space distance.

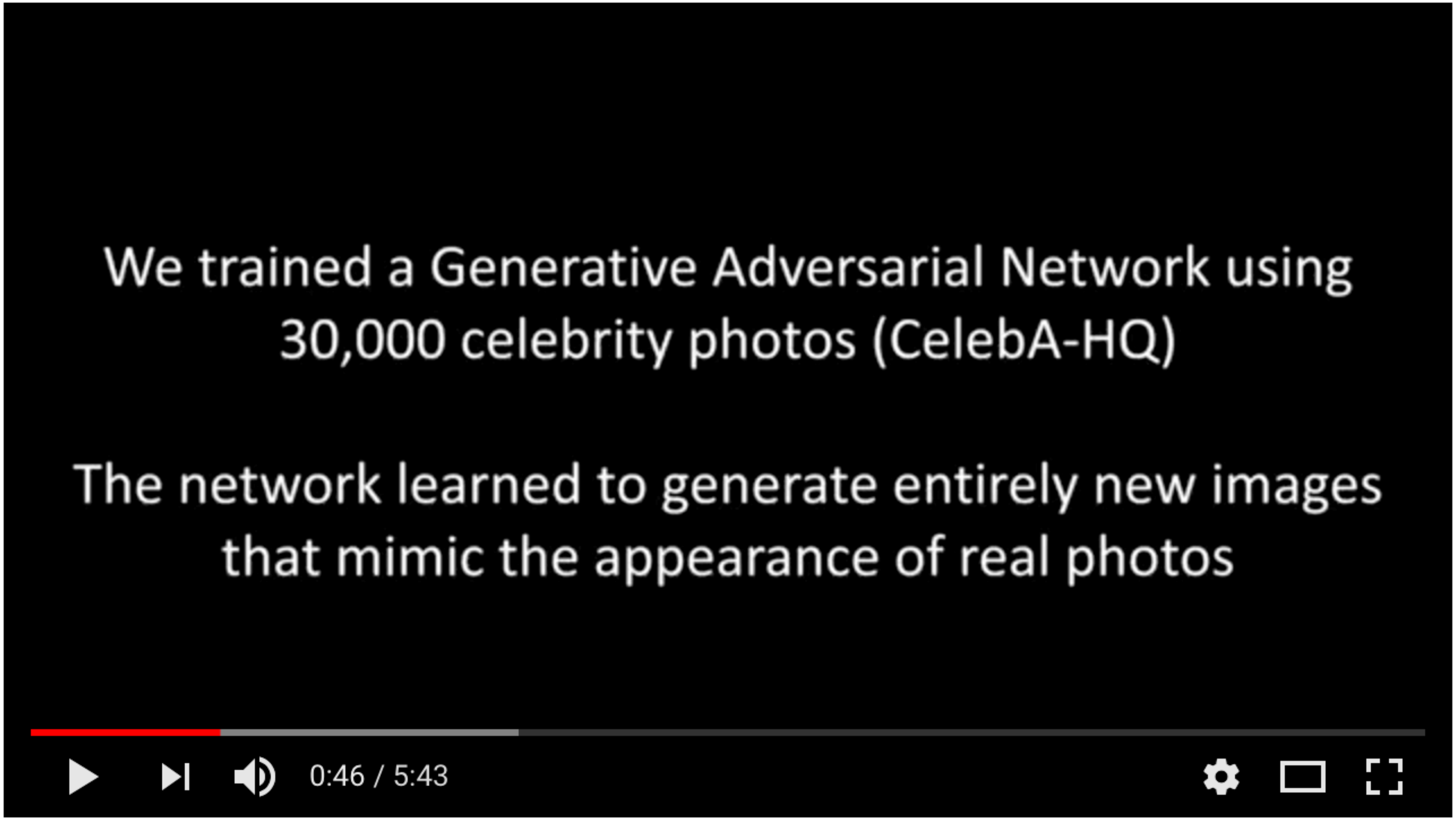
(Only the crop highlighted in bottom right image was used for comparison in order to exclude image background and focus the search on matching facial features.)

Least squares generative adversarial networks (LSGAN)

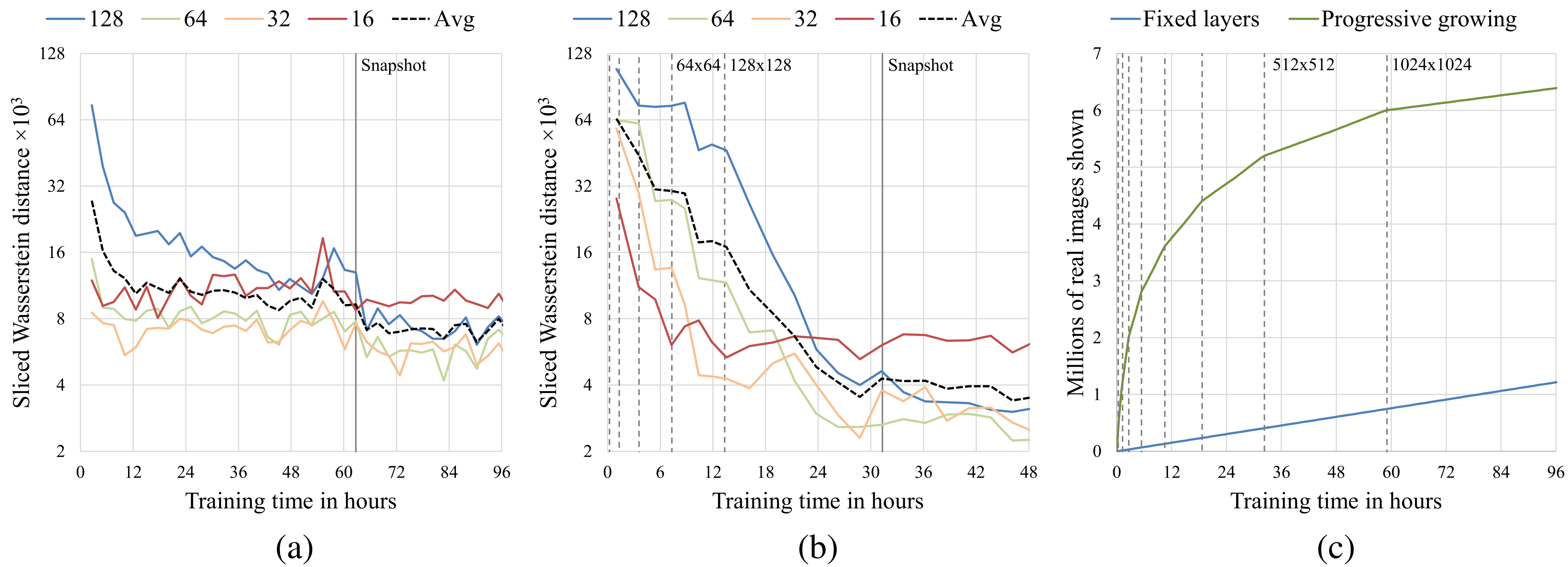
Improved training of Wasserstein GANs

Mao et al. (2016b) (128×128)   Gulrajani et al. (2017) (128×128)   Our (256×256)

Visual quality comparison in LSUN BEDROOM https://www.youtube.com/watch?v=G06dEcZ-QTg&feature=youtu.be



CONVERGENCE AND TRAINING SPEED



(On a single-GPU setup using NVIDIA Tesla P100)

(a) sliced Wasserstein distance on one level of the Laplacian pyramid, the vertical line indicates the point where we stop the training in Table 1.

(b) The vertical lines indicate points where we double the resolution of G and D.

(c) Effect of progressive growing on the raw training speed in 1024 × 1024 resolution.

Other

- Rating: 5.67
- Why? Chair says: “AnonReviewer1 (gives rating 1) has noted that the authors have revealed their names through GitHub, thus violating the double-blind submission requirement of ICLR; if not for this issue, the reviewer’s rating would have been 8.”