MIT HAN LAB

Lecture 19
Efficient Video Understanding and Generative Models

Song Han
songhan@mit.edu

Section 1

Efficient Video Understanding

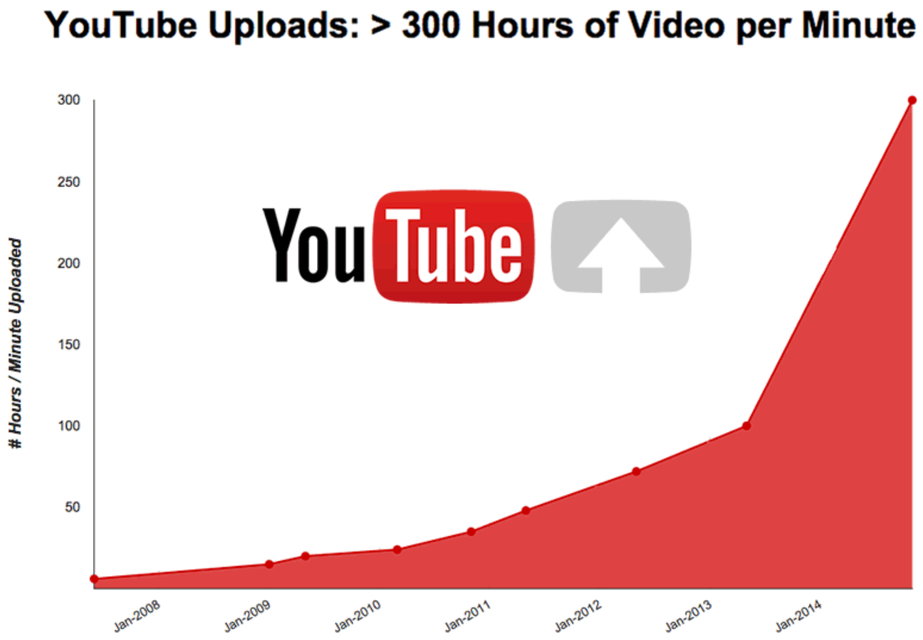
Lecture Plan

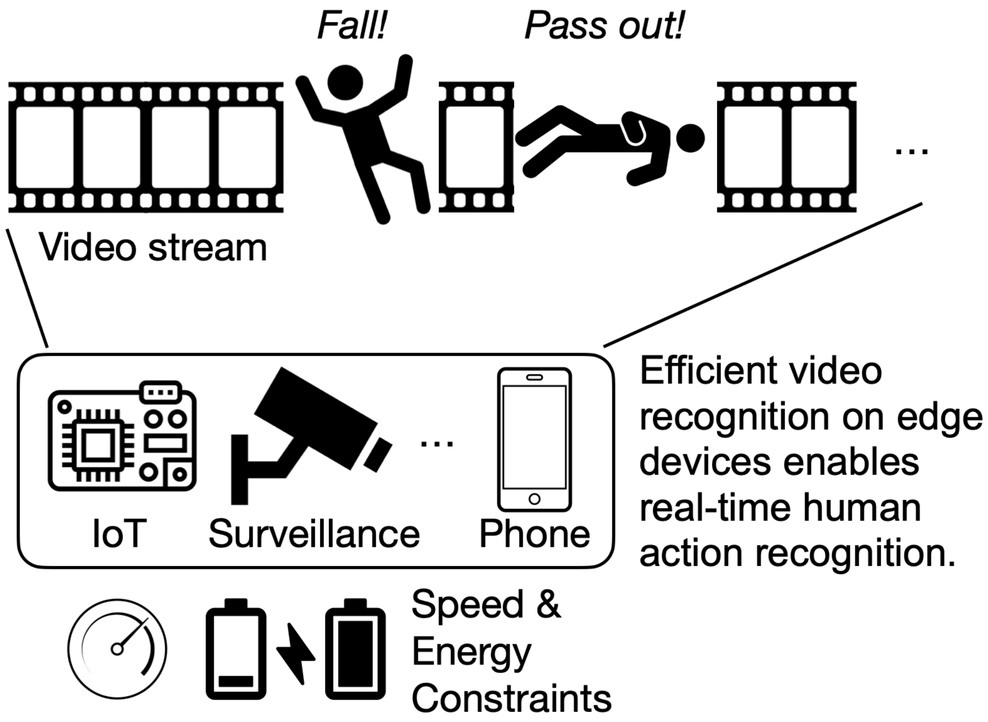
Efficient video understanding

1. 2D CNNs for video understanding
2. 3D CNNs for video understanding
3. Temporal Shift Module (TSM)
4. Other efficient methods for video understanding

Background

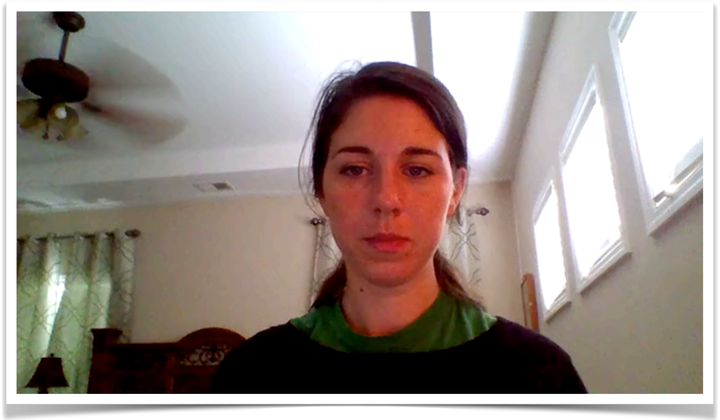
- Videos are growing explosively: 10^5 hours of videos are uploaded to YouTube per day
- Efficient Video processing is essential for both Cloud and Edge (e.g., hospitals)

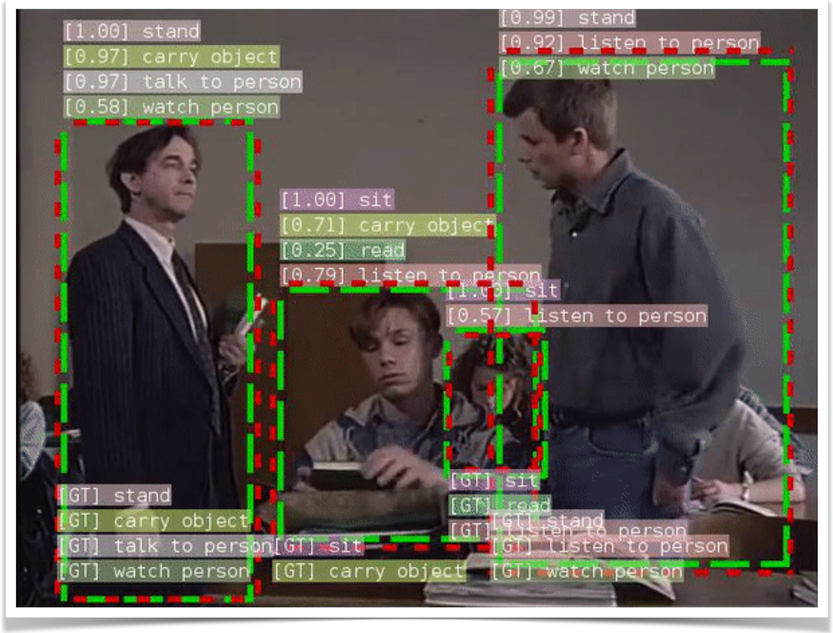


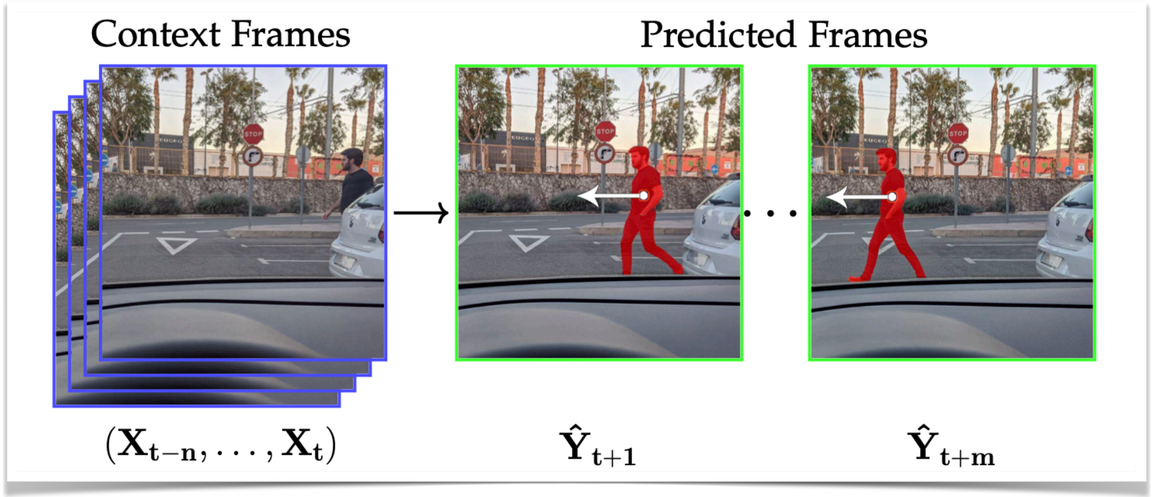


Background

- Numerous applications:
  - Action/gesture recognition
  - Scene understanding
  - Video search for ads
  - Video prediction for autonomous driving
  - …



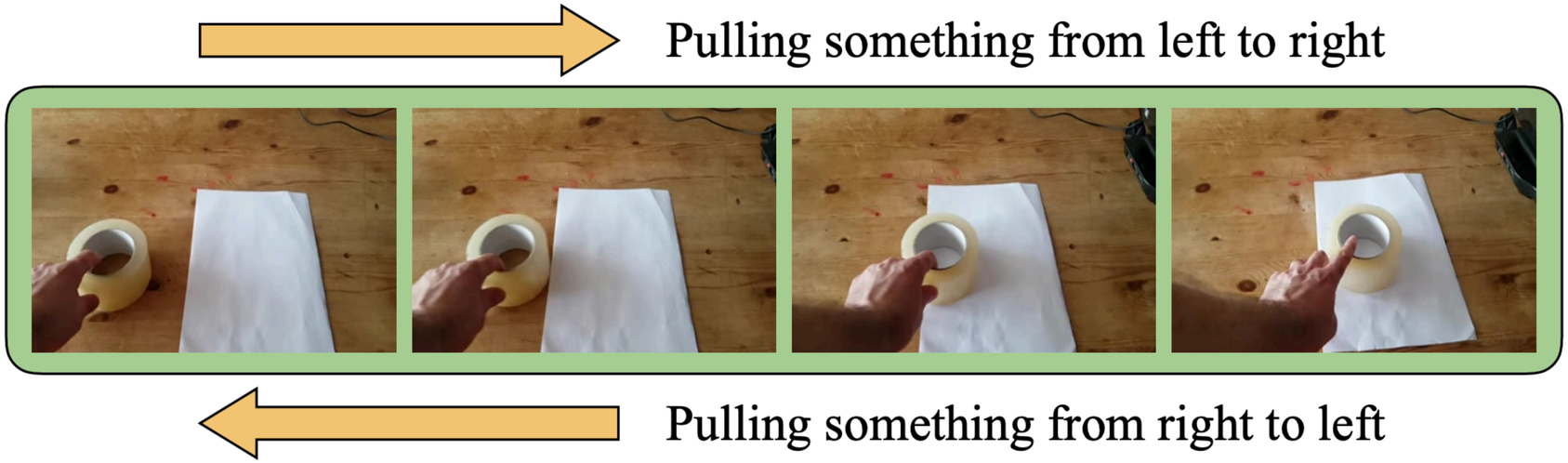




Jester dataset; https://github.com/facebookresearch/SlowFast;
A Review on Deep Learning Techniques for Video Prediction [Oprea et al., 2020]

Background

- Difference between video and image processing: **temporal** modeling



Lecture Plan

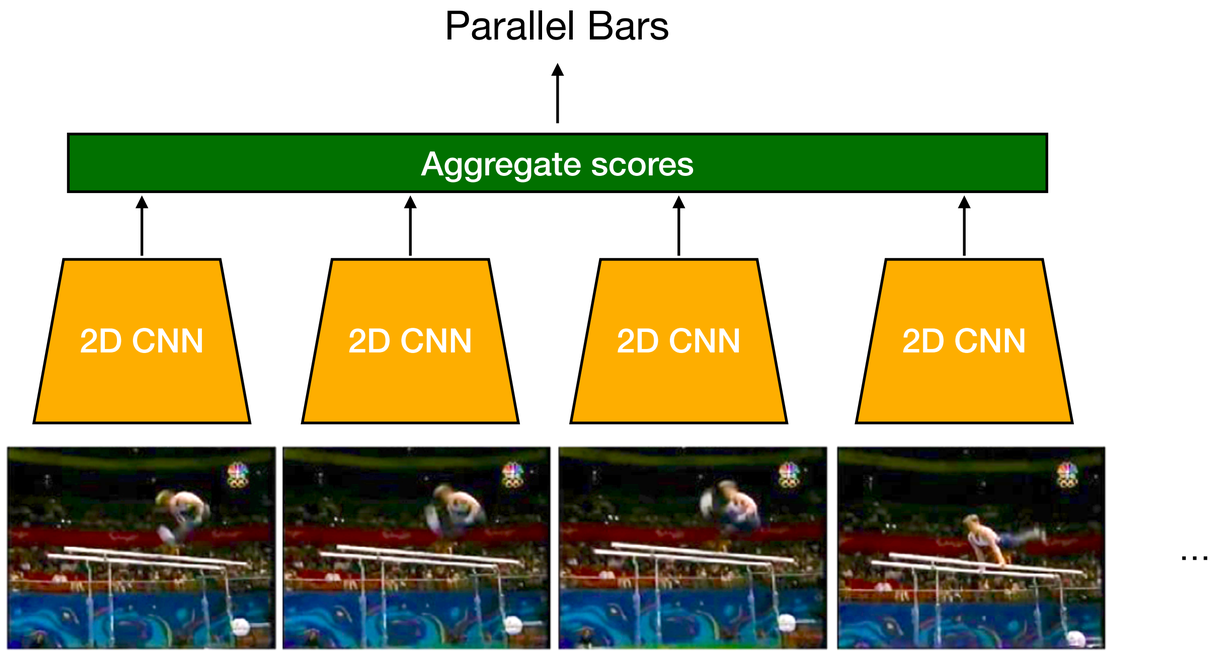
Efficient video understanding

1. **2D CNNs for video understanding**
2. 3D CNNs for video understanding
3. Temporal Shift Module (TSM)
4. Other efficient methods for video understanding

Existing Work on Video Understanding

2D CNN-based method

- Sample frames and process with 2D CNNs (just like images)
- Aggregate scores for prediction (e.g., average, max, …)

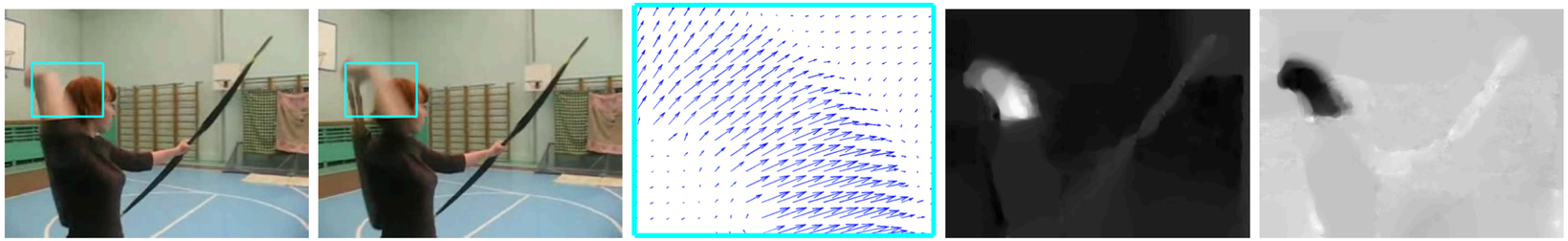


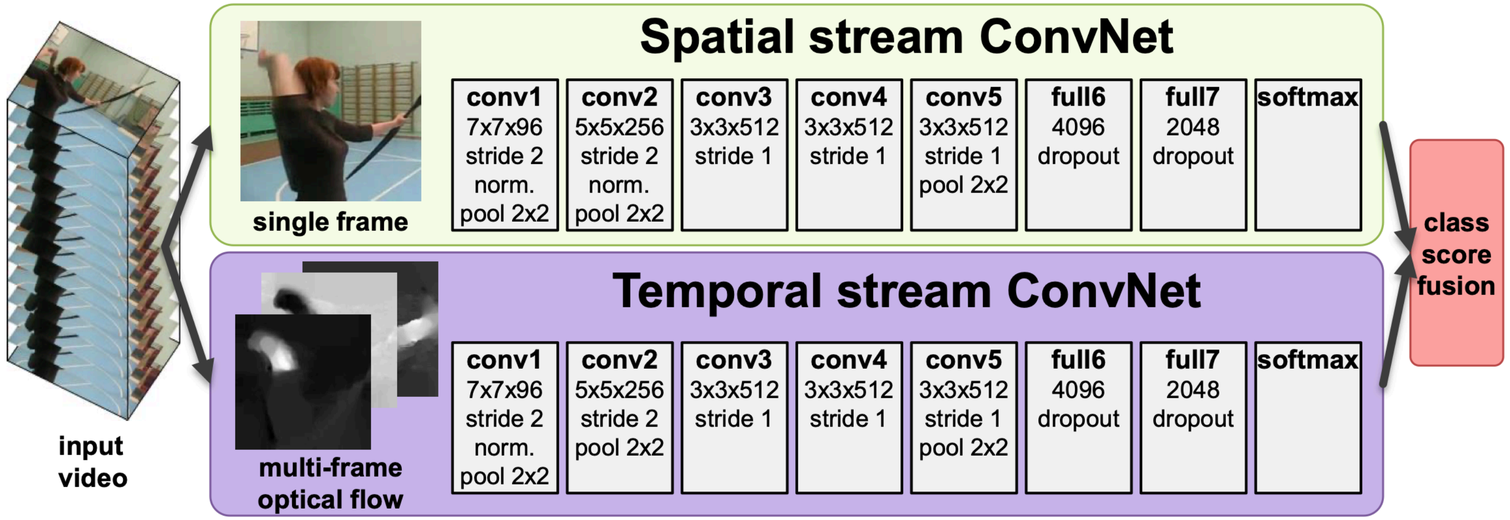
Temporal Segment Networks: Towards Good Practices for Deep Action Recognition [Wang et al. 2016]

Existing Work on Video Understanding

2D CNN-based method

- Two steam method: spatial + temporal (optical flow, **slow** to compute)



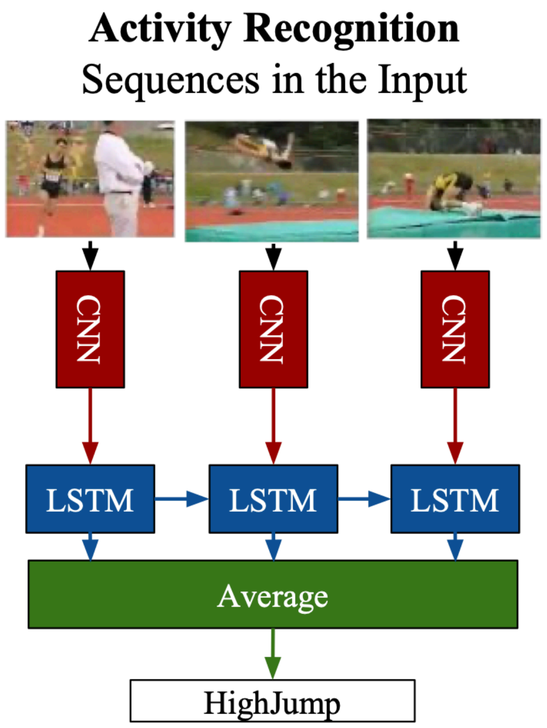


Two-Stream Convolutional Networks for Action Recognition in Videos [Simonyan et al., 2014]

Existing Work on Video Understanding

2D CNN-based method

- 2D CNN + Post-fusion (e.g., LSTM)
- Low-level information is lost during the backbone inference



Long-term Recurrent Convolutional Networks for Visual Recognition and Description [Donahue et al., 2016]

Existing Work on Video Understanding

2D CNN-based method

- **Pro**:
  - Compute-efficient. Reuse 2D CNNs from image recognition.
- **Con**:
  - Aggregating 2D CNNs cannot model temporal information. Low accuracy on video benchmarks.
  - Optical flow is very slow to compute (much slower than the deep network itself).
  - Late fusion cannot model low-level temporal relationships.

Lecture Plan

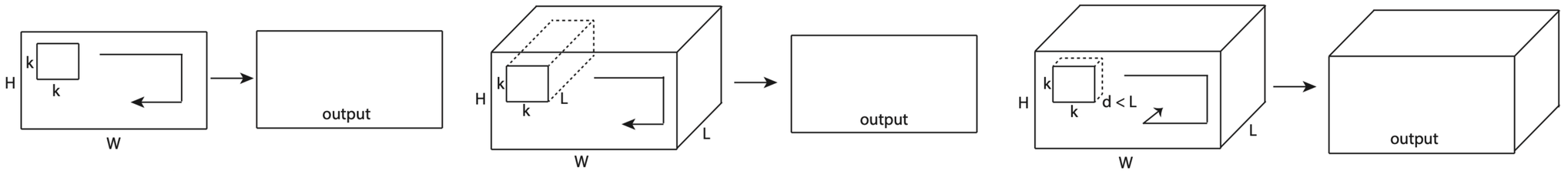
Efficient video understanding

1. 2D CNNs for video understanding
2. **3D CNNs for video understanding**
3. Temporal Shift Module (TSM)
4. Other efficient methods for video understanding

Existing Work on Video Understanding

3D CNN-based method

- **C3D**: Convolve not only on the spatial dimensions but also on temporal dimensions
- Jointly model spatiotemporal information



(a) 2D convolution (b) 2D convolution on multiple frames (c) 3D convolution

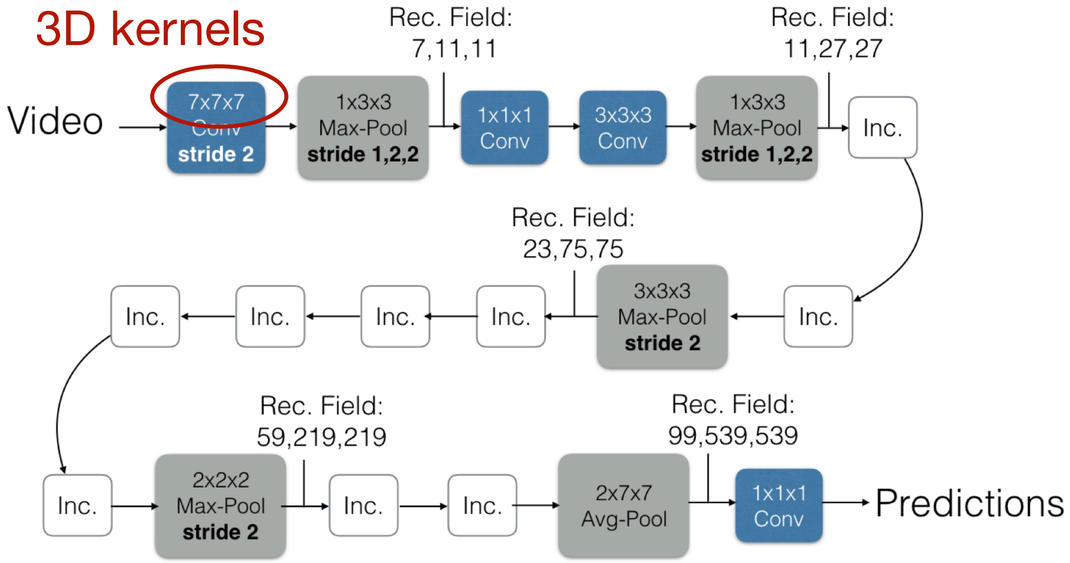
Learning Spatiotemporal Features with 3D Convolutional Networks [Tran et al. 2014]

Existing Work on Video Understanding

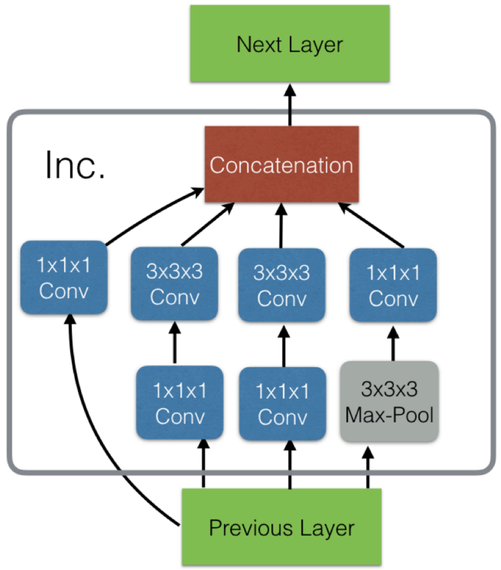
3D CNN-based method

- 3D CNNs have more parameters; less data-efficient
- **I3D**: initialize 3D CNNs with 2D CNN weights (e.g., pre-trained on ImageNet) by inflation!
  - Repeat the weights across temporal dimension for initialization

Inflated Inception-V1



Inception Module (Inc.)



Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset [Carreira et al. 2018]

Existing Work on Video Understanding

3D CNN-based method

- **Pros**:
  - Jointly modeling spatiotemporal information
  - Can model low-, middle-, high-level information
- **Cons**:
  - Large cost (model size, computation) due to the extra temporal dimension

Existing Work on Video Understanding

3D CNN-based method

- **Pros**:
  - Jointly modeling spatiotemporal information
  - Can model low-, middle-, high-level information
- **Cons**:
  - Large cost (model size, computation) due to the extra temporal dimension

- **Question**: can we achieve 3D CNN performance at 2D CNN cost?

Lecture Plan

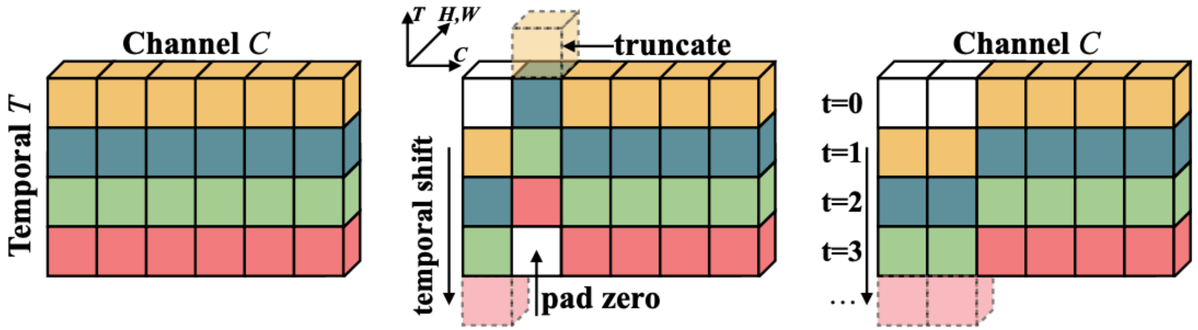
Efficient video understanding

1. 2D CNNs for video understanding
2. 3D CNNs for video understanding
3. **Temporal Shift Module (TSM)**
4. Other efficient methods for video understanding

Temporal Shift Module (TSM)

Zero FLOPs, zero parameter for temporal modeling

- **Bi-directional** TSM shifts part of the channels along the temporal dimension to facilitate information exchange among neighboring frames.
- **Uni-directional** TSM shifts channels from past to future for online video understanding.
- It can be inserted into *off-the-shelf* 2D CNN to enable temporal modeling at the cost of *zero FLOPs* and *zero parameters*



(a) The original tensor without shift.

(b) Offline temporal shift (bi-direction).

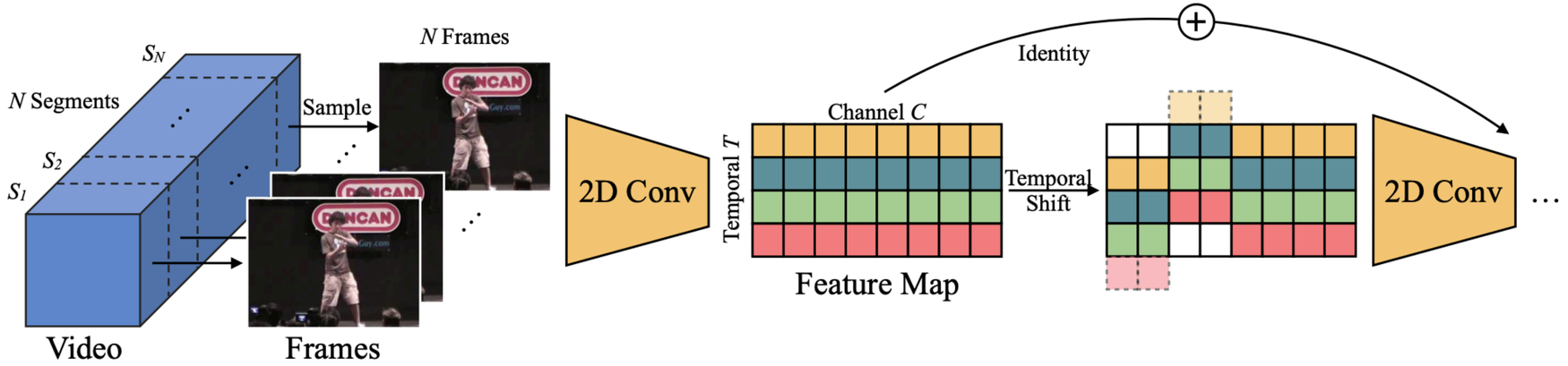
(c) Online temporal shift (uni-direction).

TSM: Temporal Shift Module for Efficient Video Understanding [Lin et al., 2019]

Temporal Shift Module (TSM)

Inserting TSM into 2D CNNs

- **Offline** TSM models
  - Application: action recognition, fall detection, video recommendation, etc.
  - Frames are processed in a batch

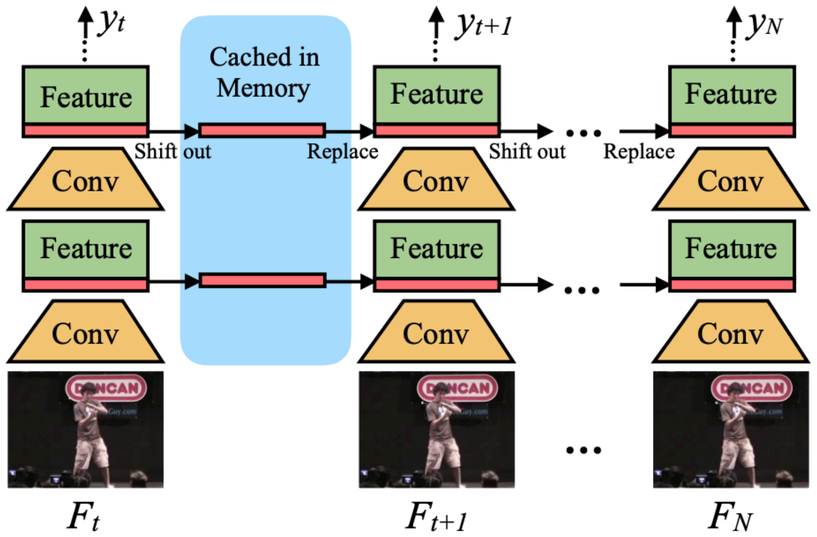


TSM: Temporal Shift Module for Efficient Video Understanding [Lin et al., 2019]

Temporal Shift Module (TSM)

Inserting TSM into 2D CNNs

- **Online** TSM models
  - Application: autonomous driving, improving detection with temporal cues, etc.
  - Frame-by-frame processing, streaming fashion.



TSM: Temporal Shift Module for Efficient Video Understanding [Lin et al., 2019]

Temporal Shift Module (TSM)

Simple implementation with few lines of code

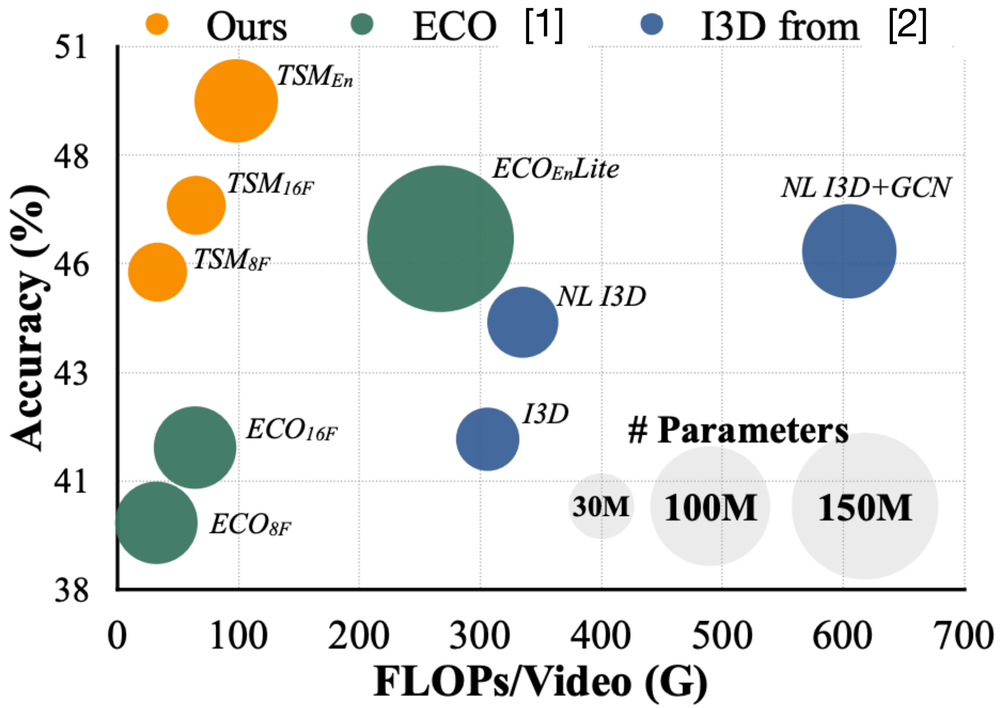
```
# shape of x: [N, T, C, H, W]
out = torch.zeros_like(x)
fold = c // fold_div
out[:, :-1, :fold] = x[:, 1:, :fold]  # shift left
out[:, 1:, fold: 2 * fold] = x[:, :-1, fold: 2 * fold]  # shift right
out[:, :, 2 * fold:] = x[:, :, 2 * fold:]  # not shift
return out
```

TSM: Temporal Shift Module for Efficient Video Understanding [Lin et al., 2019]

Temporal Shift Module (TSM)

High performance and high accuracy

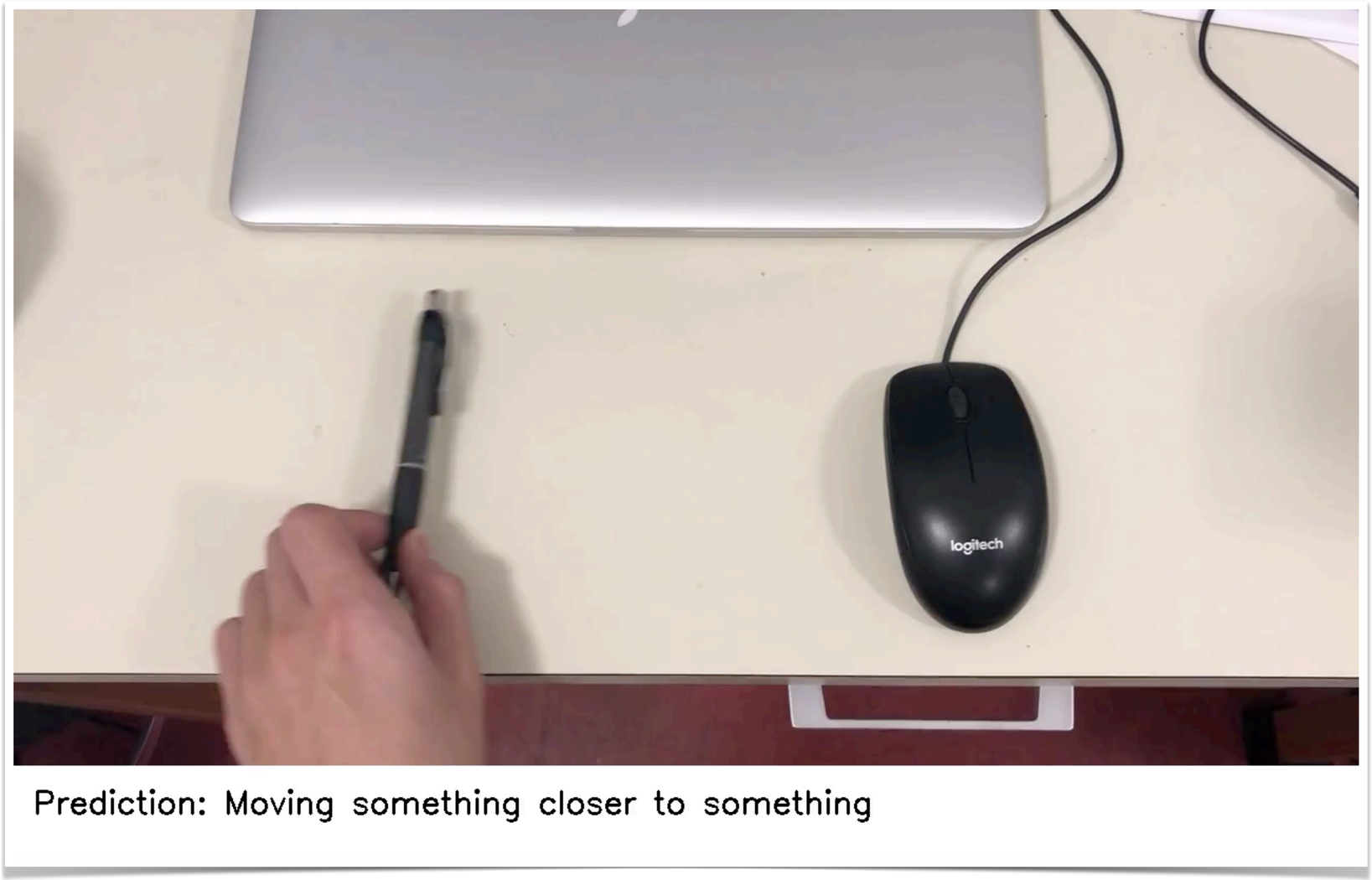
- It consumes **3×** less computation than the ECO family, **6×** less computation than the Non-local I3D family while achieving better performance on Something-Something dataset



[1] Efficient Convolutional Network for Online Video Understanding [Zolfaghari et al., 2018]
[2] Non-local Neural Networks [Wang et al., 2017]

Temporal Shift Module (TSM)

Real-life demos: Something-Something relationship



Temporal Shift Module (TSM)

Lower latency, higher accuracy

- Batch size=**1**. Measured on NVIDIA Tesla P100. Each **row** represents a video.

I3D:
Latency: **164.3** ms/Video Something-V1 Acc.: **41.6**%

TSM:
Latency: **17.4** ms/Video Something-V1 Acc.: **43.4**%

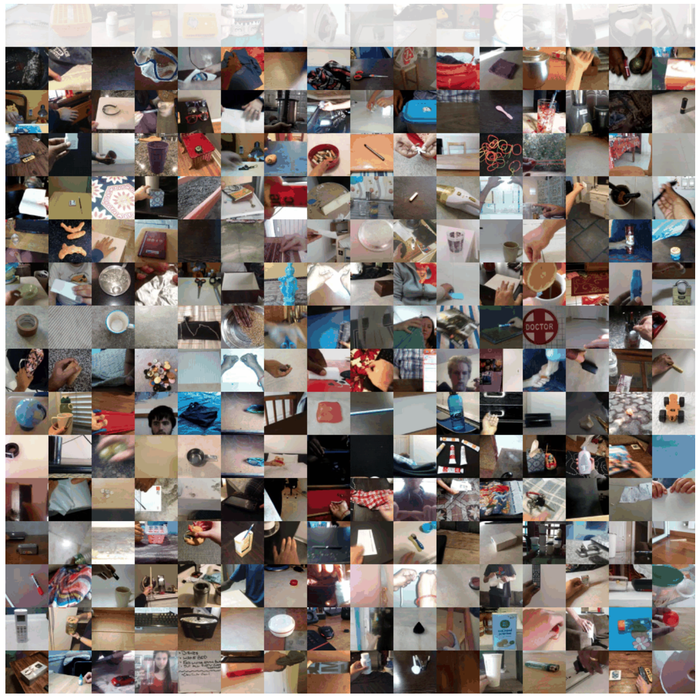
Speed-up: 9x

Temporal Shift Module (TSM)

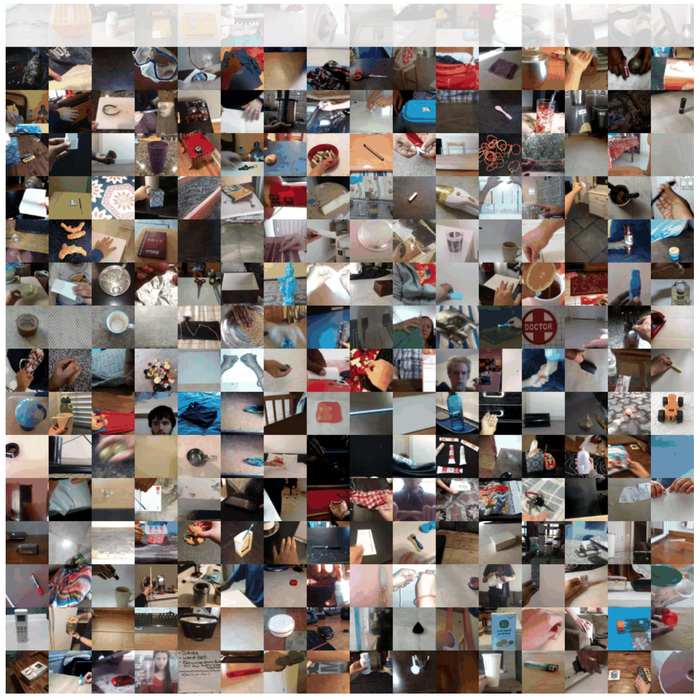
Higher throughput, higher accuracy

- Batch size=**16**. Measured on NVIDIA Tesla P100. Each **square** represents a video.

I3D:
Throughput: **6.1** video/s
Something-V1 Acc.: **41.6**%



TSM:
Throughput: **77.4** video/s
Something-V1 Acc.: **43.4**%



12.7x larger throughput

Temporal Shift Module (TSM)

Online Video Recognition

- Online video recognition process the video frame-by-frame in a streaming manner

Table 5. Comparing the accuracy of offline TSM and online TSM on different datasets. Online TSM brings negligible latency overhead.

Model	Latency	Kinetics	UCF101	HMDB51	Something
TSN	8.5ms	70.7%	91.7%	65.1%	20.4%
+Offline	-	74.7%	96.0%	73.2%	45.5%
+Online	9.1ms	74.8%	95.5%	73.6%	44.3%

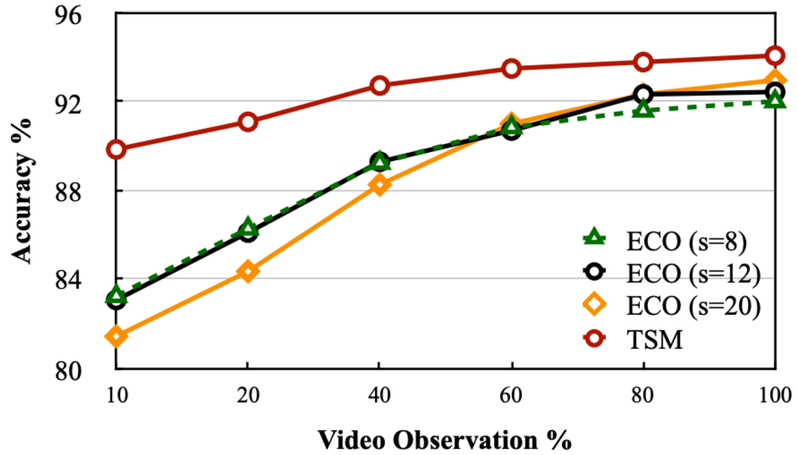


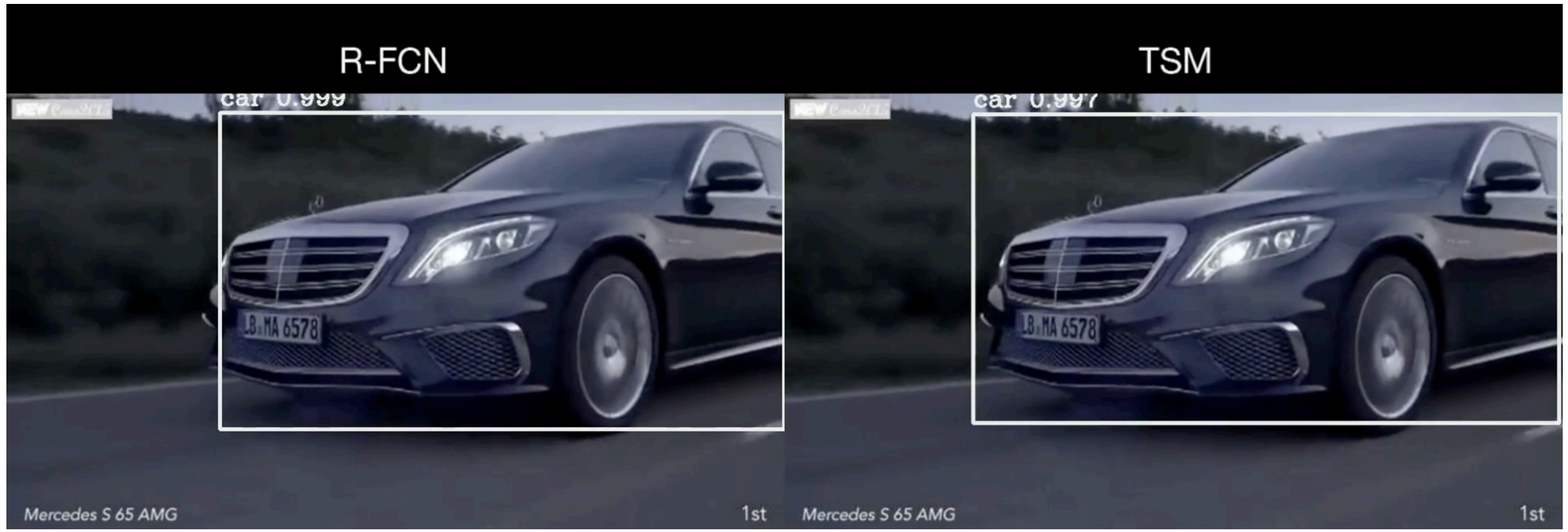
Figure 7. Early recognition on UCF101. TSM gives high prediction accuracy after only observing a small portion of the video.

Temporal Shift Module (TSM)

Improve online video recognition

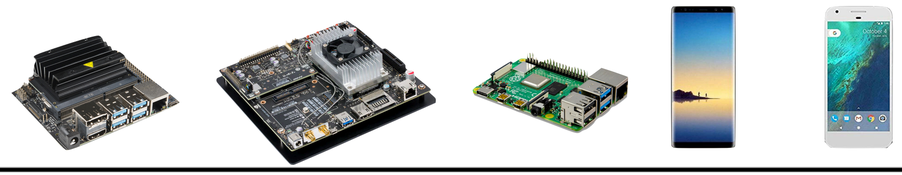
- TSM can improve 2D CNN-based online recognition with the help of temporal information

Model	Online	Need Flow	Latency	mAP			
				Overall	Slow	Medium	Fast
R-FCN [2]	✓		1×	74.7	83.6	72.5	51.4
FGFA [4]		✓	2.5×	75.9	**84.0**	74.4	55.6
Online TSM	✓		1×	**76.3**	83.4	**74.8**	**56.0**



Temporal Shift Module (TSM)

Scaling down: low-latency low-power deployment



Devices	Jetson Nano		Jetson TX2		Rasp.	Note8	Pixel1
	CPU	GPU	CPU	GPU			
FPS	20.9	74.6	27.5	117.6	14.4	29.0	21.1
Power (watt)	4.8	4.5	5.6	5.8	3.8	-	-

LED Bulb Level!

Temporal Shift Module (TSM)

***Scaling up*: large-scale video training**

- Scaling up TSM to SUMMIT super computer (world No. 1 in 2019)

SUMMIT Super Computer:

- CPU: 2 x 16 Core IBM POWER9 (connected via dual NVLINK bricks, 25GB/s each side)
- GPU: 6 x NVIDIA Tesla V100
- RAM: 512 GB DDR4 memory
- Data Storage: HDD
- Connection: Dual-rail EDR InfiniBand network of 23 GB/s

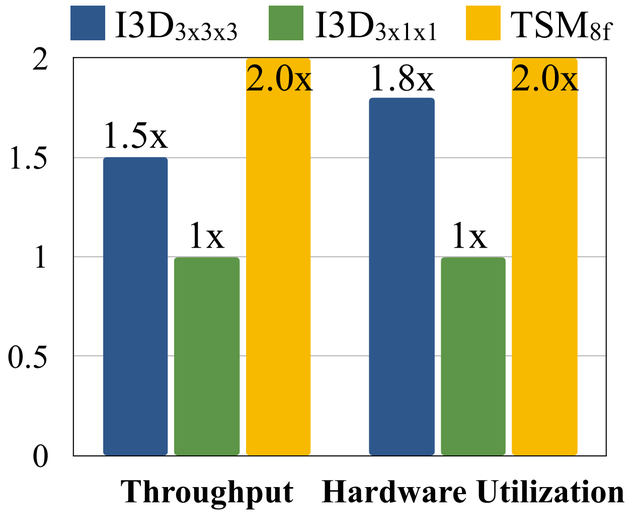
Acknowledgment: IBM and Oak Ridge National Lab

Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]

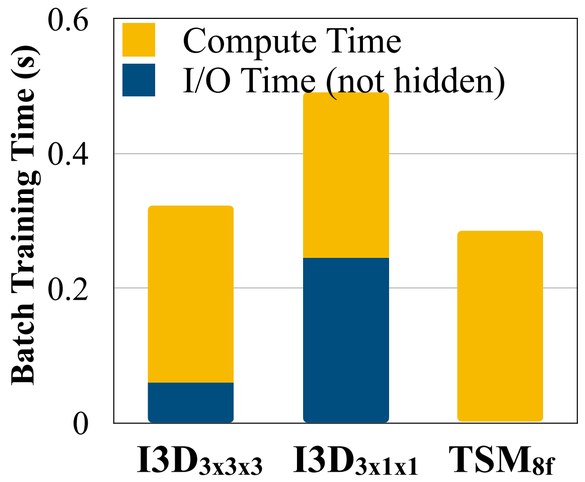
Temporal Shift Module (TSM)

Key to faster training: less FLOPs, I/O and communication

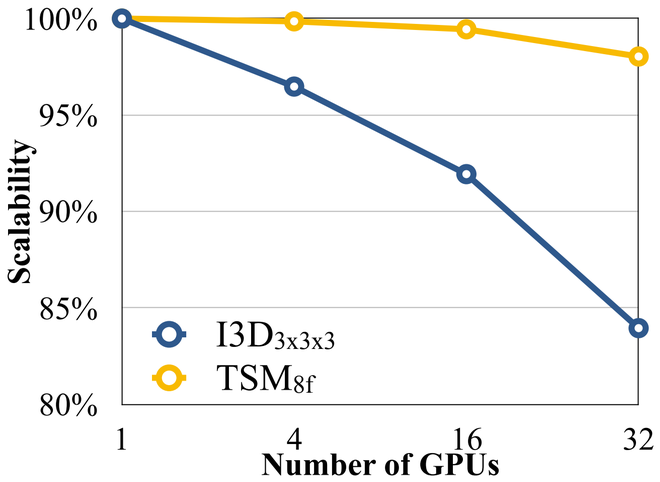
Computation | **Data movement**



(a) TSM has fewer FLOPs, better throughput and utilization.



(b) TSM is I/O light, decreasing the total batch time.



(c) TSM has better scalability due to smaller model size.

throughput: samples/s
utilization: FLOP/s

Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]

Temporal Shift Module (TSM)

Scaling up: large-scale video training

- We are able to speed up the training by **200x**, from **2 days to 14 minutes**.
- Model setup: 8-frame ResNet-50 TSM for video recognition
- Dataset: Kinetics (240k training videos) x 100 epoch

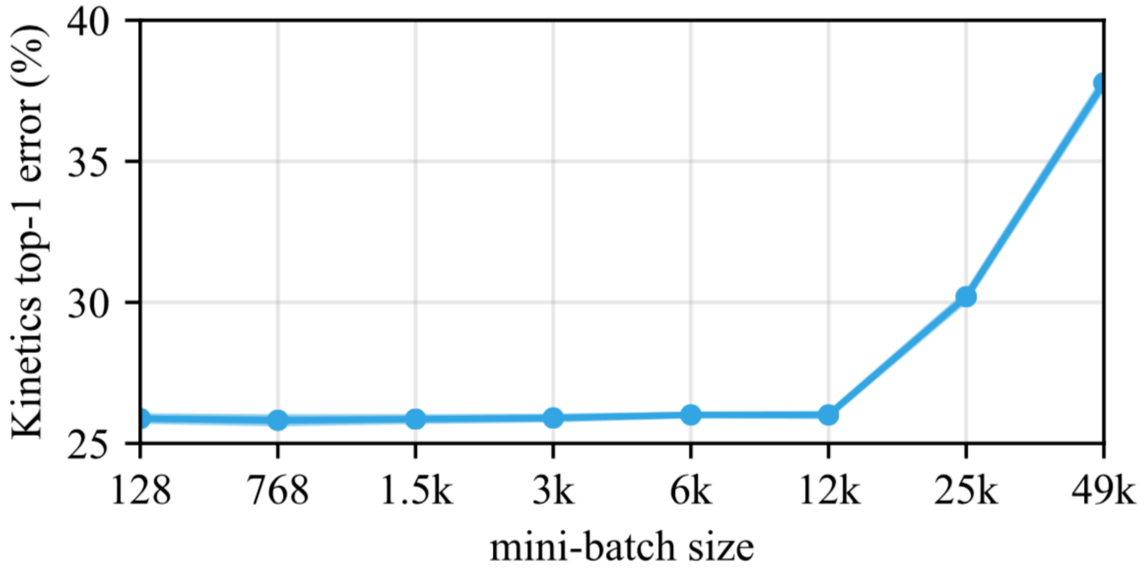
	Training Time	Accuracy	Peak GPU Performance	Speed-up
1 SUMMIT Nodes (6 GPUs)	**49h 50min**	74.1%	46.5TFLOP/s	Theoretical: 128x Actual: 106x
128 SUMMIT Nodes (768 GPUs)	**28min**	74.1%	5,989TFLOP/s	Theoretical: 256x Actual: 211x
256 SUMMIT Nodes (1536 GPUs)	**14min**	74.0%	11,978TFLOP/s	

Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]

Temporal Shift Module (TSM)

Scaling up: accuracy vs. batch size

- The performance of TSM model does not degrade when we scale up the mini-batch size to 12k (videos)

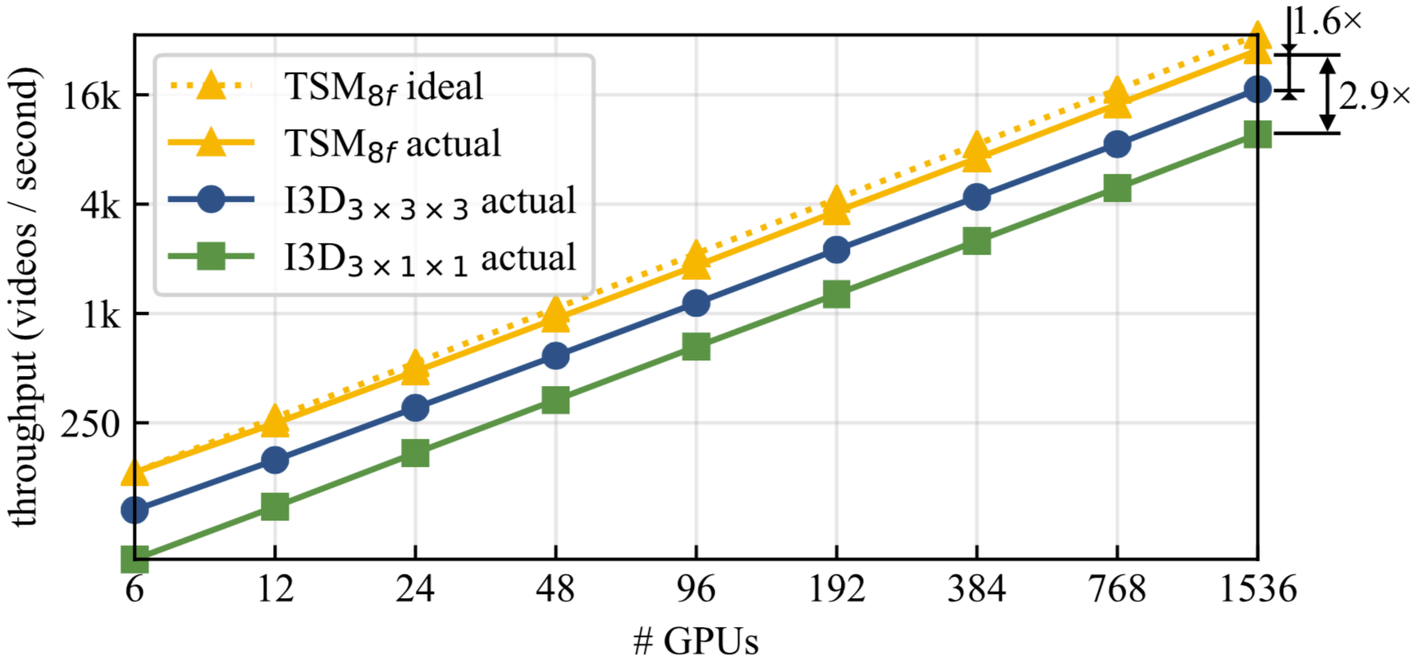


Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]

Temporal Shift Module (TSM)

Scaling up: scalability vs. models

- TSM model achieves 1.6x and 2.9x higher training throughput compared to previous I3D models



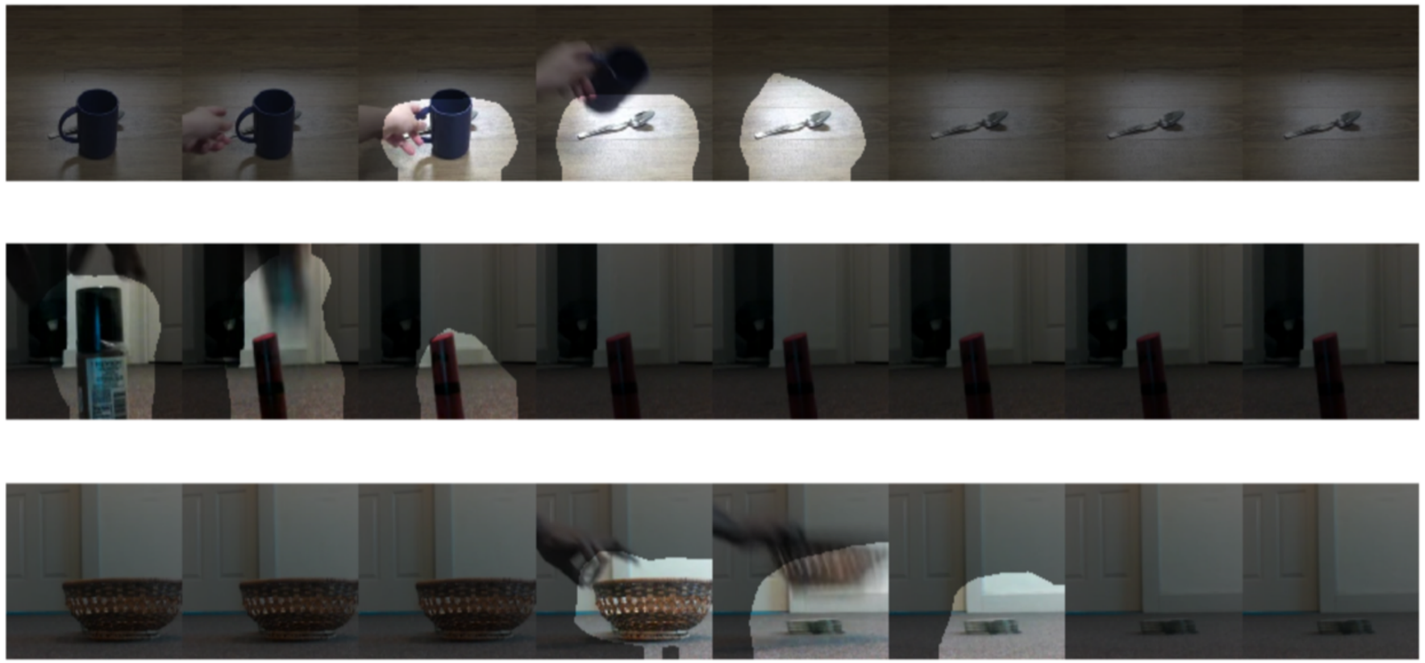
Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]

Temporal Shift Module (TSM)

Dissecting TSM

- Spatiotemporal localization emerges from the training on classification tasks
- Each channel learns different semantics

Channel 5: Move something away



Temporal Shift Module (TSM)

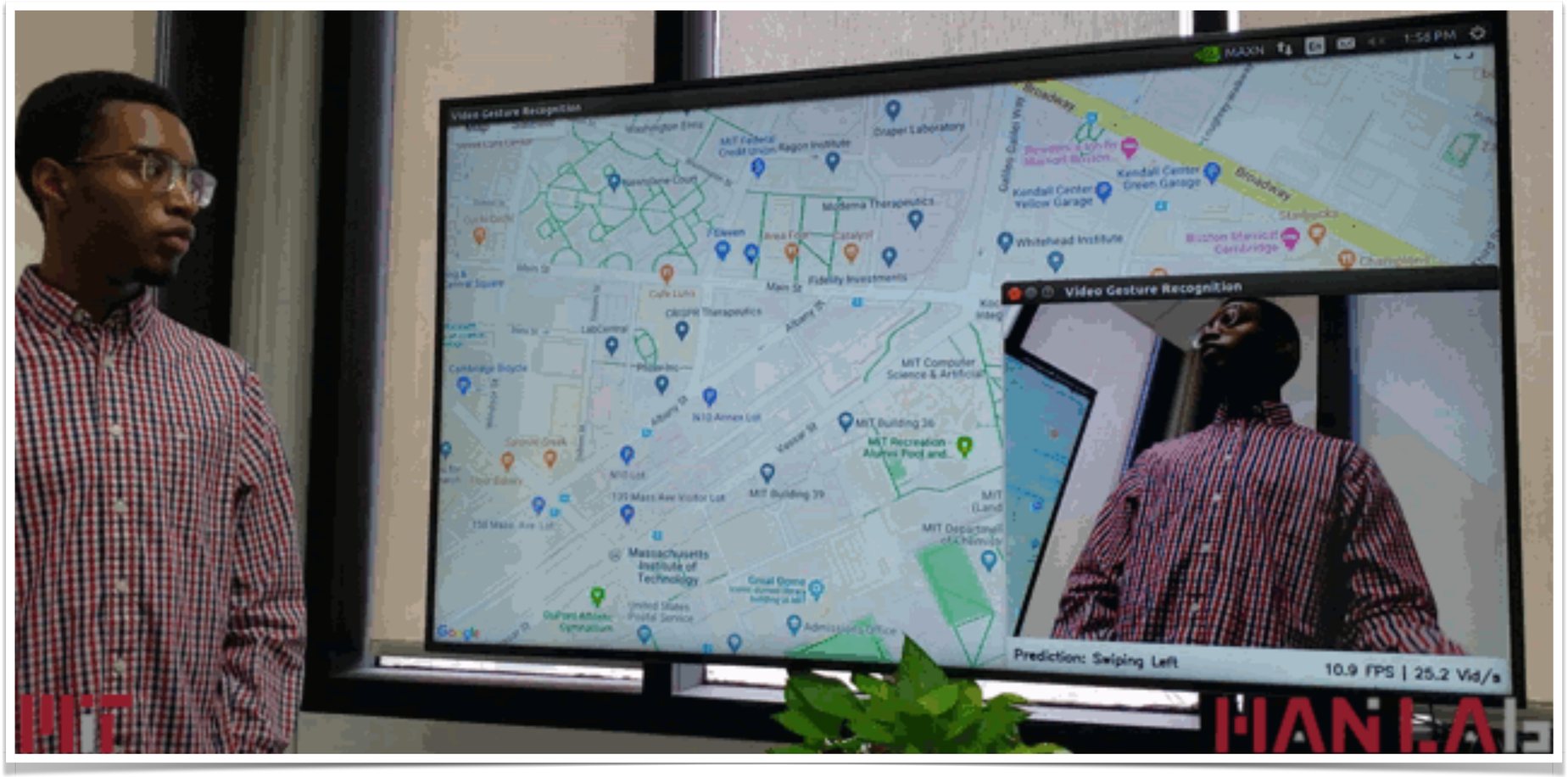
Dissecting TSM

- Spatiotemporal localization emerges from the training on classification tasks
- Each channel learns different semantics

Channel 446: Push to left

Temporal Shift Module (TSM)

Real-life demos: Google Map navigation with gestures



Lecture Plan

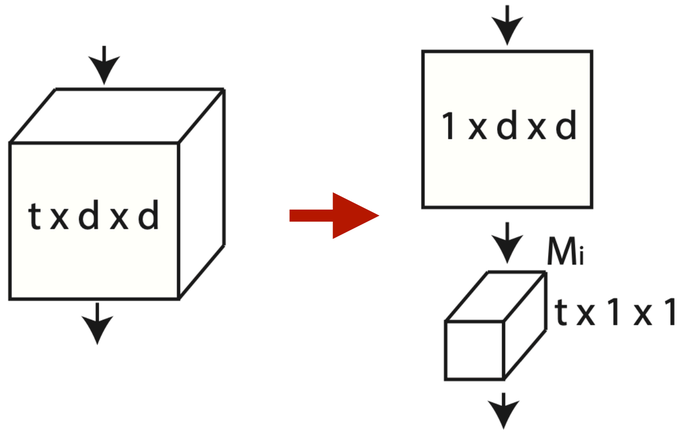
Efficient video understanding

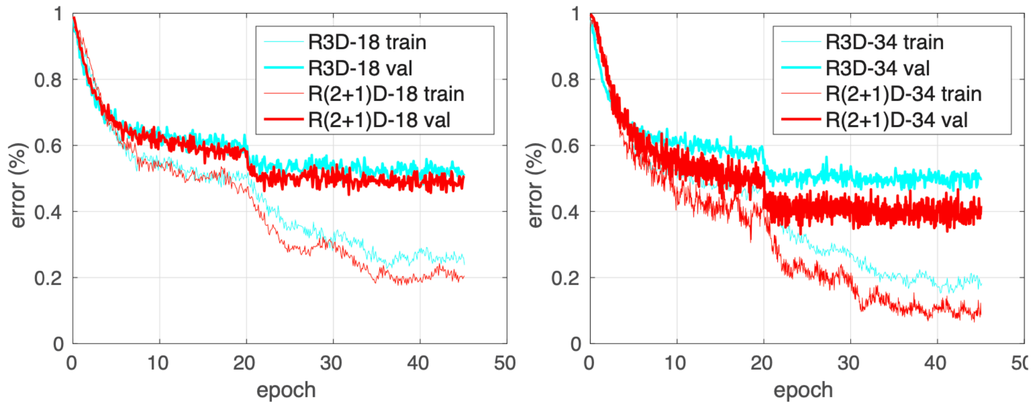
1. 2D CNNs for video understanding
2. 3D CNNs for video understanding
3. Temporal Shift Module (TSM)
4. **Other efficient methods for video understanding**

Other Efficient Methods

Kernel Decomposition

- Decompose 3D CNN kernels into 2D spatial + 1D temporal
- Lighter, easier to train than the 3D counterpart



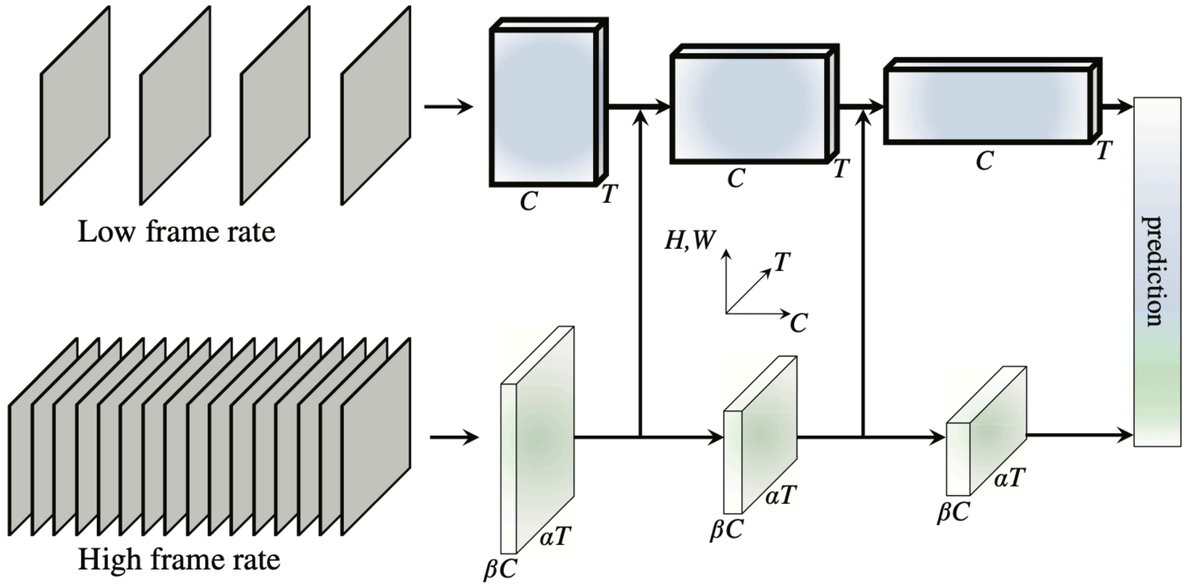


A Closer Look at Spatiotemporal Convolutions for Action Recognition [Tran et al., 2018]

Other Efficient Methods

Multi-scale modeling

- Large spatial resolution + large temporal resolution -> Too heavy!
- Instead, use a **two-branch** design:
  - **Small** spatial resolution + **large** temporal resolution
  - **Large** spatial resolution + **small** temporal resolution

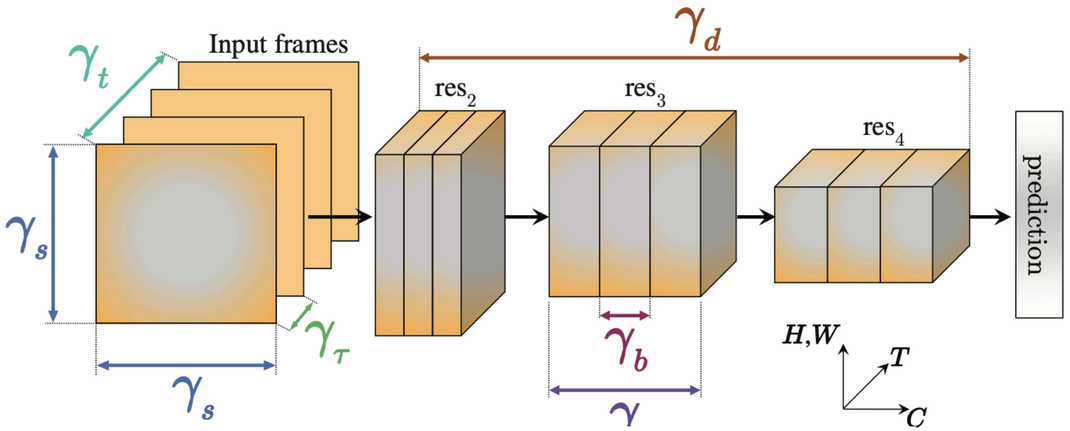


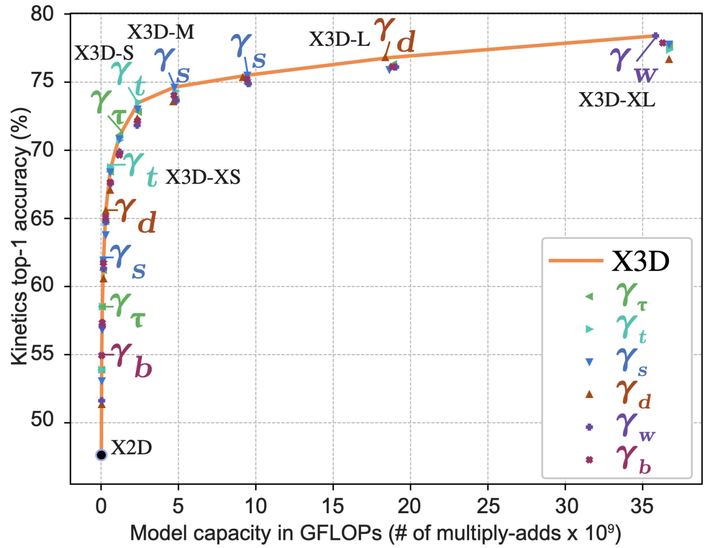
SlowFast Networks for Video Recognition [Feichtenhofer et al., 2019]

Other Efficient Methods

Neural architecture search

- Neural architecture search for 3D CNNs by gradually expanding:
  - Temporal duration γt,
  - frame rate γτ
  - spatial resolution γs
  - width γw
  - bottleneck width γb
  - and depth γd



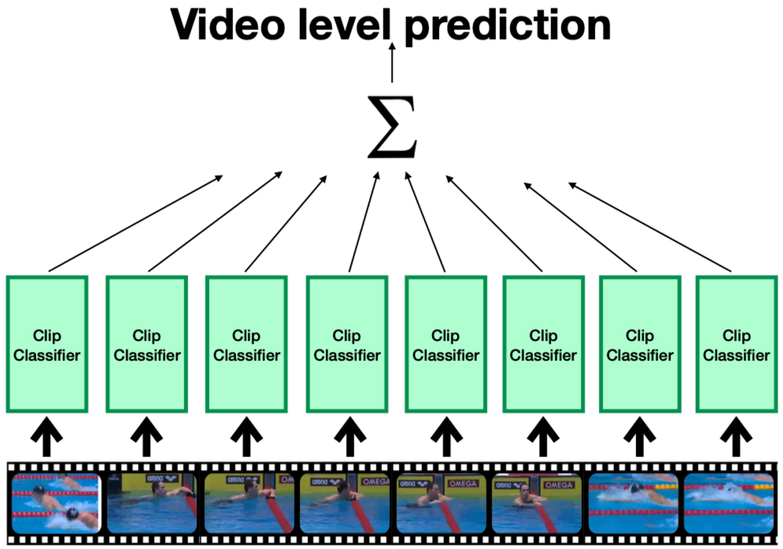


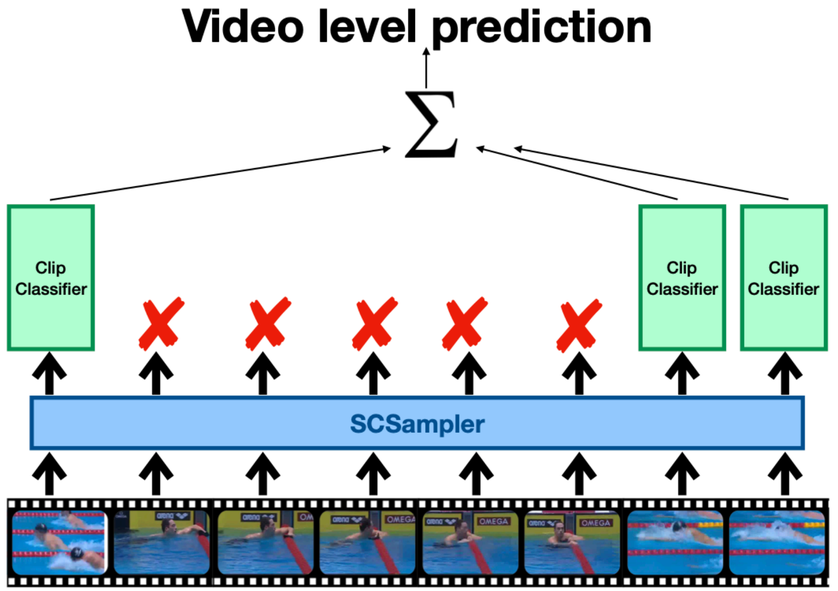
X3D: Expanding Architectures for Efficient Video Recognition [Feichtenhofer et al., 2020]

Other Efficient Methods

Skipping redundant frames/clips

- In a long video, usually only part of the segments are informative
- We can skip the non-salient segments to reduce computation



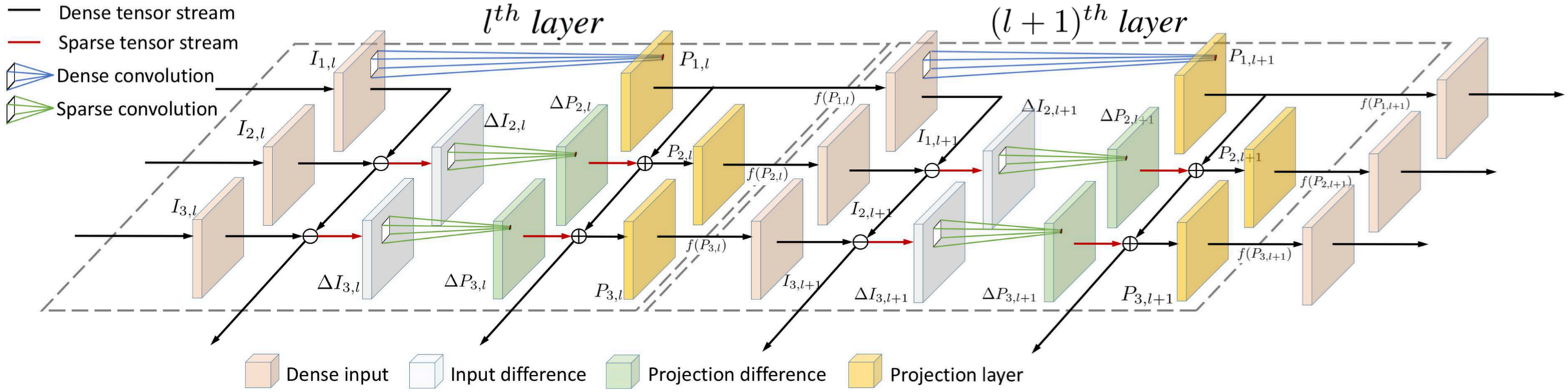


SCSampler: Sampling Salient Clips from Video for Efficient Action Recognition [Korbar et al., 2019]

Other Efficient Methods

Utilizing spatial redundancy across frames

- In a video, neighboring frames might be highly similar (sometimes almost static, like surveillance)
- Utilize the linearity of conv, and sparsity the difference:
  - $y_{t+1} = F(x_{t+1}) = F(x_t + x_{t+1} - x_t) = F(x_t + \Delta) = F(x_t) + F(\Delta) = y_t + F(\Delta)$
- Needs specialized hardware to exploit fine-grained sparsity

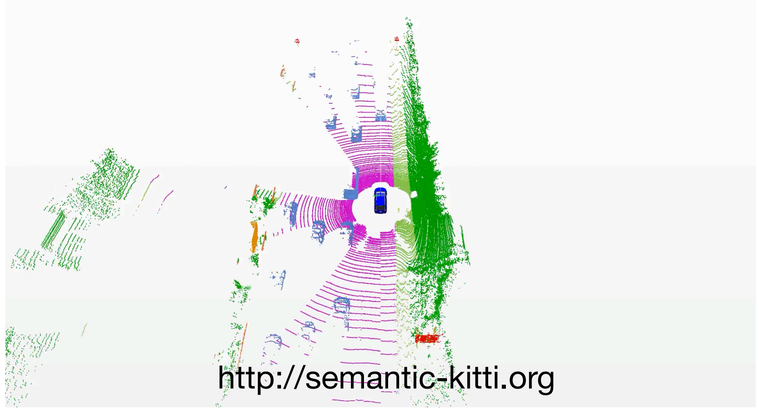


Recurrent Residual Module for Fast Inference in Videos [Pan et al., 2018]

Section 2

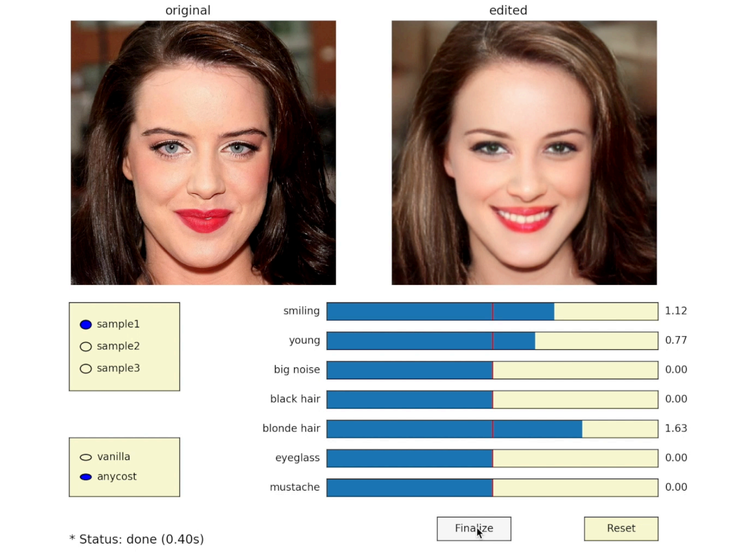
Efficient Generative Models

Application-Specific Optimizations for Efficient AI Computing

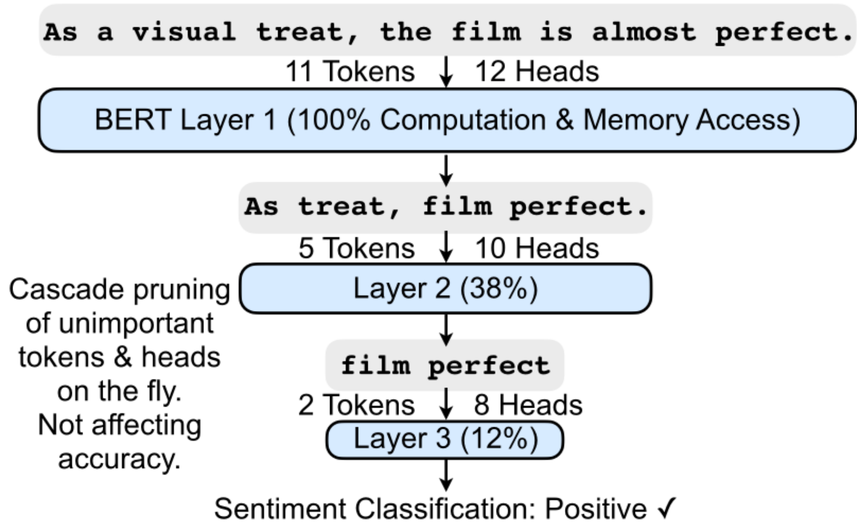


Point cloud: 3D spatial redundancy

Videos: temporal redundancy



GANs: 2D spatial redundancy (this lecture)



NLP: token redundancy

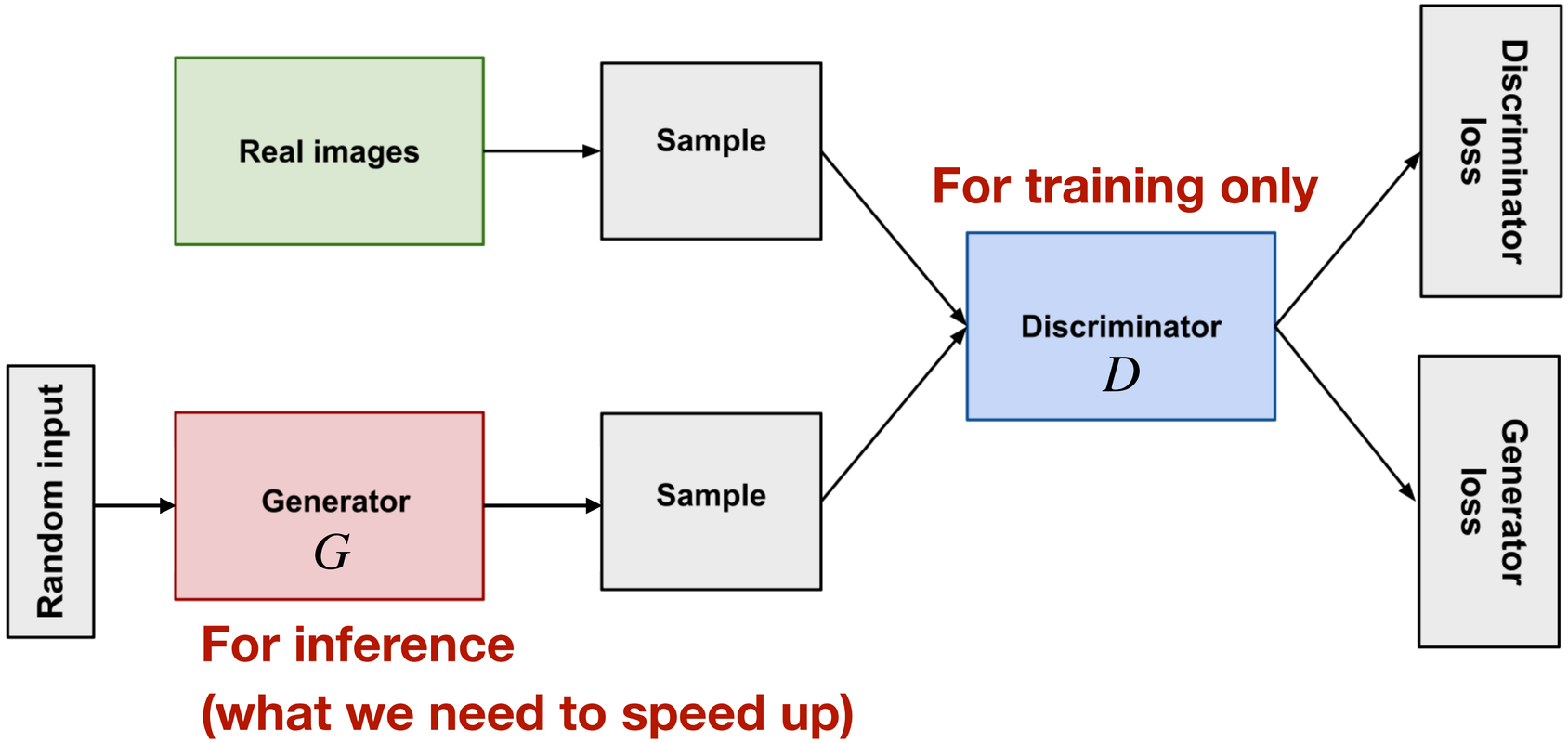
Background

Generative Adversarial Networks (GANs)

- Learning to generate by adversarial learning

real samples fake samples

$$E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))]$$



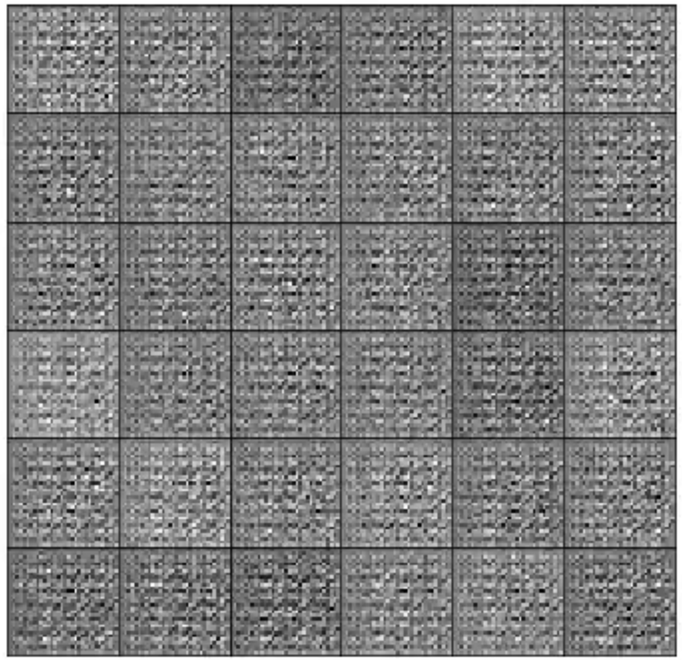
Generative Adversarial Networks [Goodfellow et al., 2014]
https://developers.google.com/machine-learning/gan/gan_structure

Background

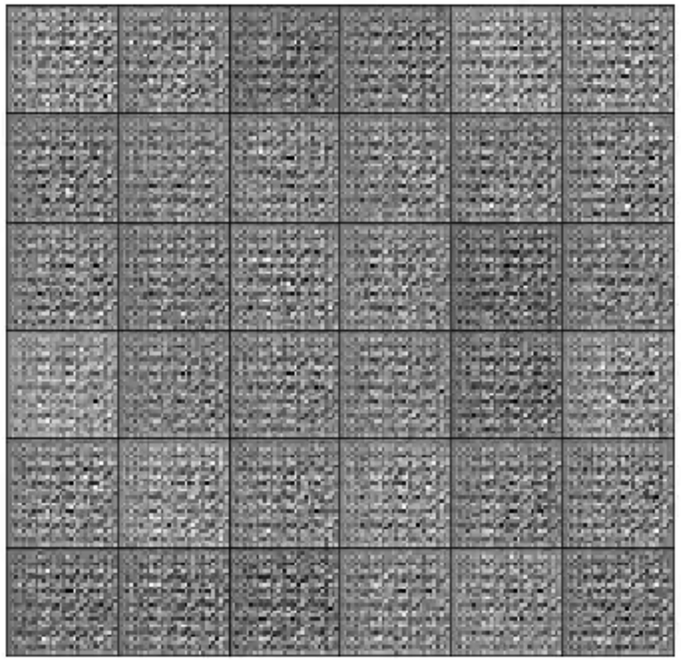
Generative Adversarial Networks (GANs)

- Example: learning to generate digits from MNIST

random initialization



training

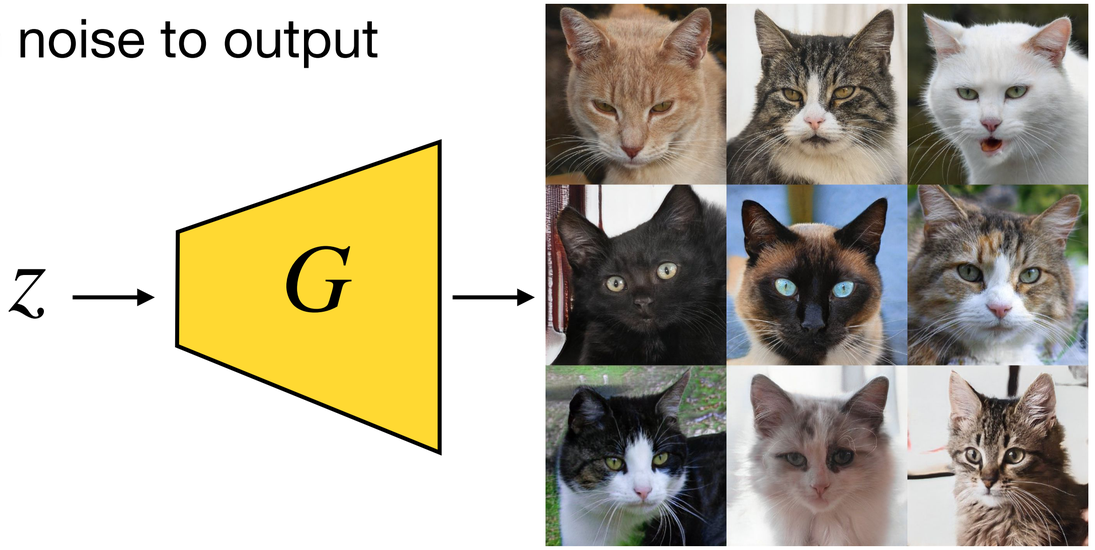


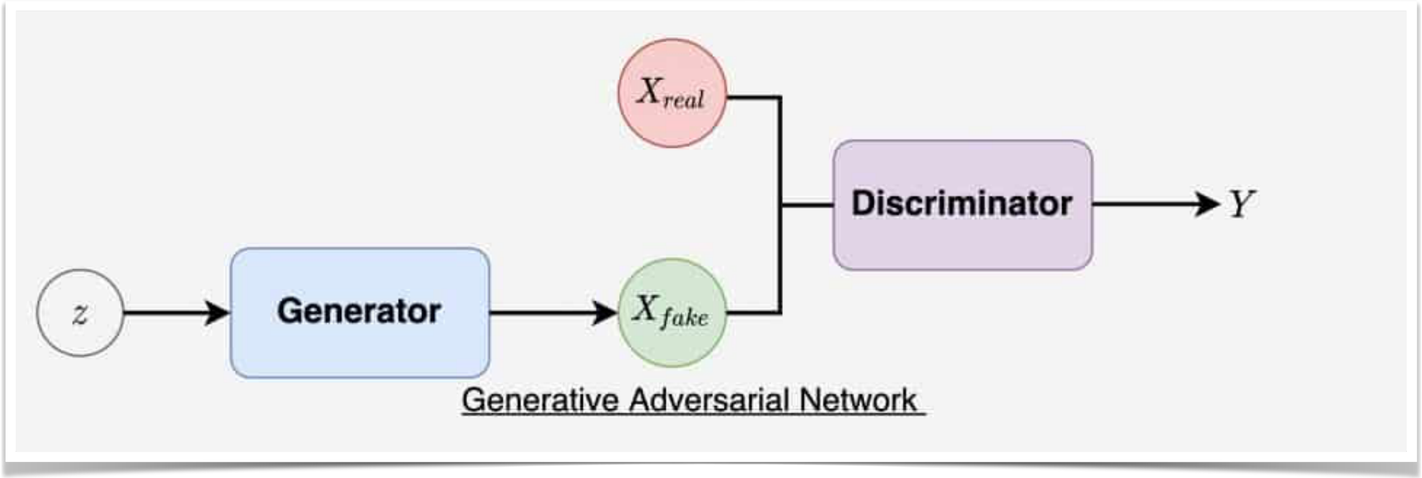
https://sthalles.github.io/intro-to-gans/

Background

Conditional vs. unconditional

- Unconditional GAN: random noise to output



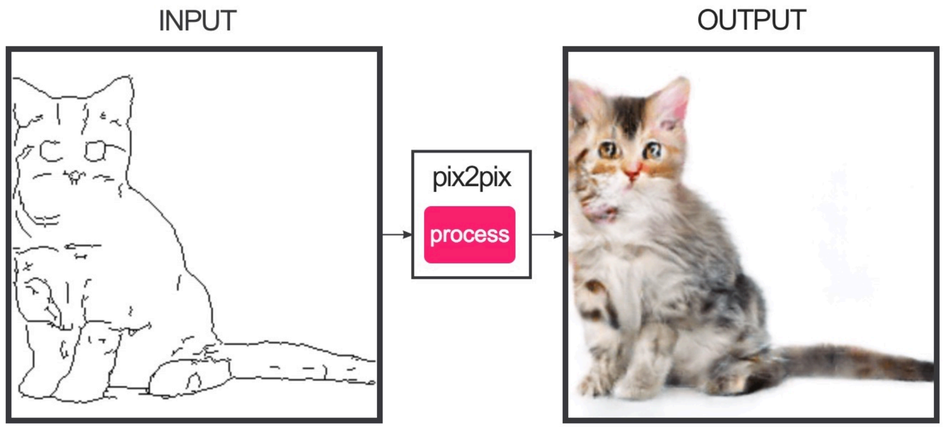


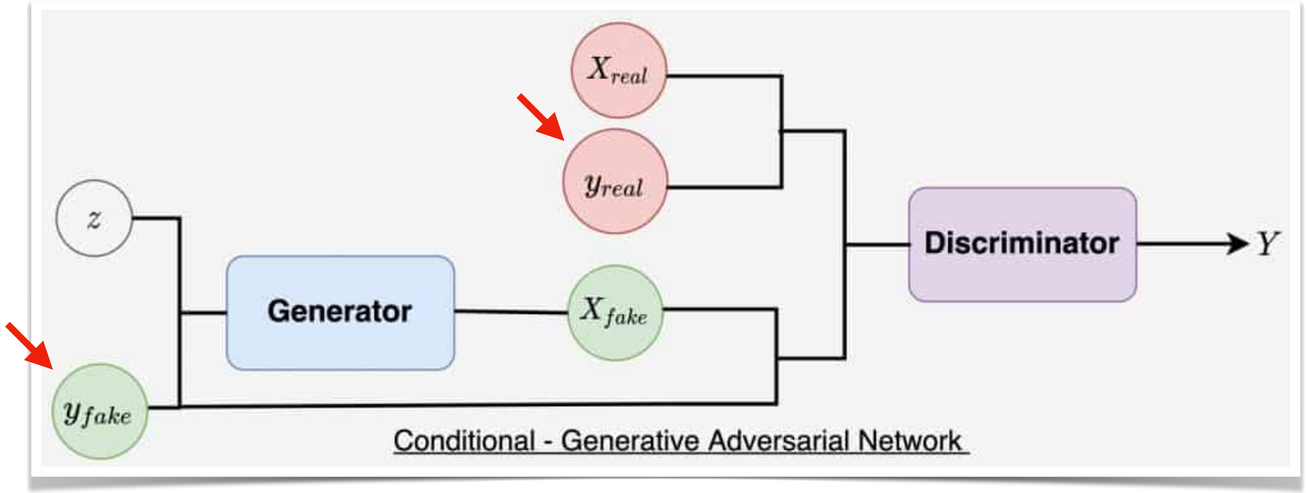
https://learnopencv.com/conditional-gan-cgan-in-pytorch-and-tensorflow/

Background

Conditional vs. unconditional

- Conditional GANs: D and G are conditioned on provided **labels**
  - class label
  - segmentation map
  - strokes
  - …



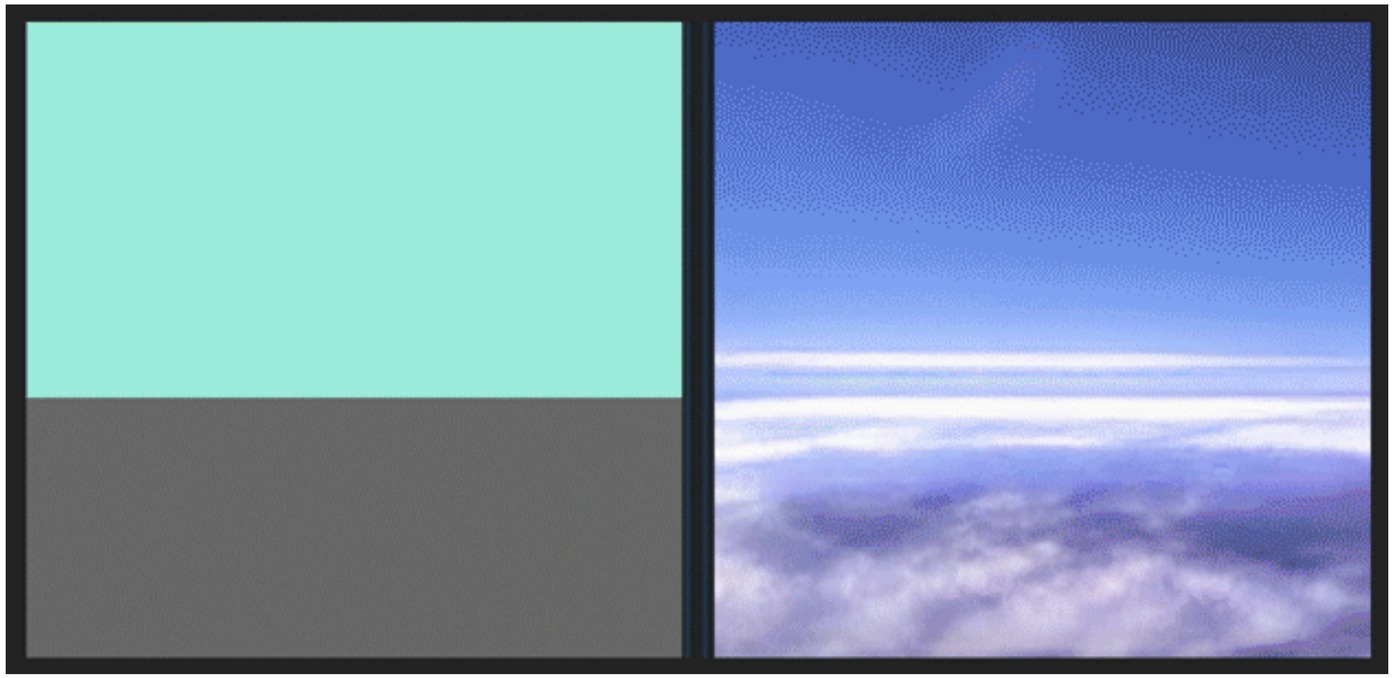


https://learnopencv.com/conditional-gan-cgan-in-pytorch-and-tensorflow/

Background

Generative models for content creation

- Conditional GANs (GauGAN) for drawing

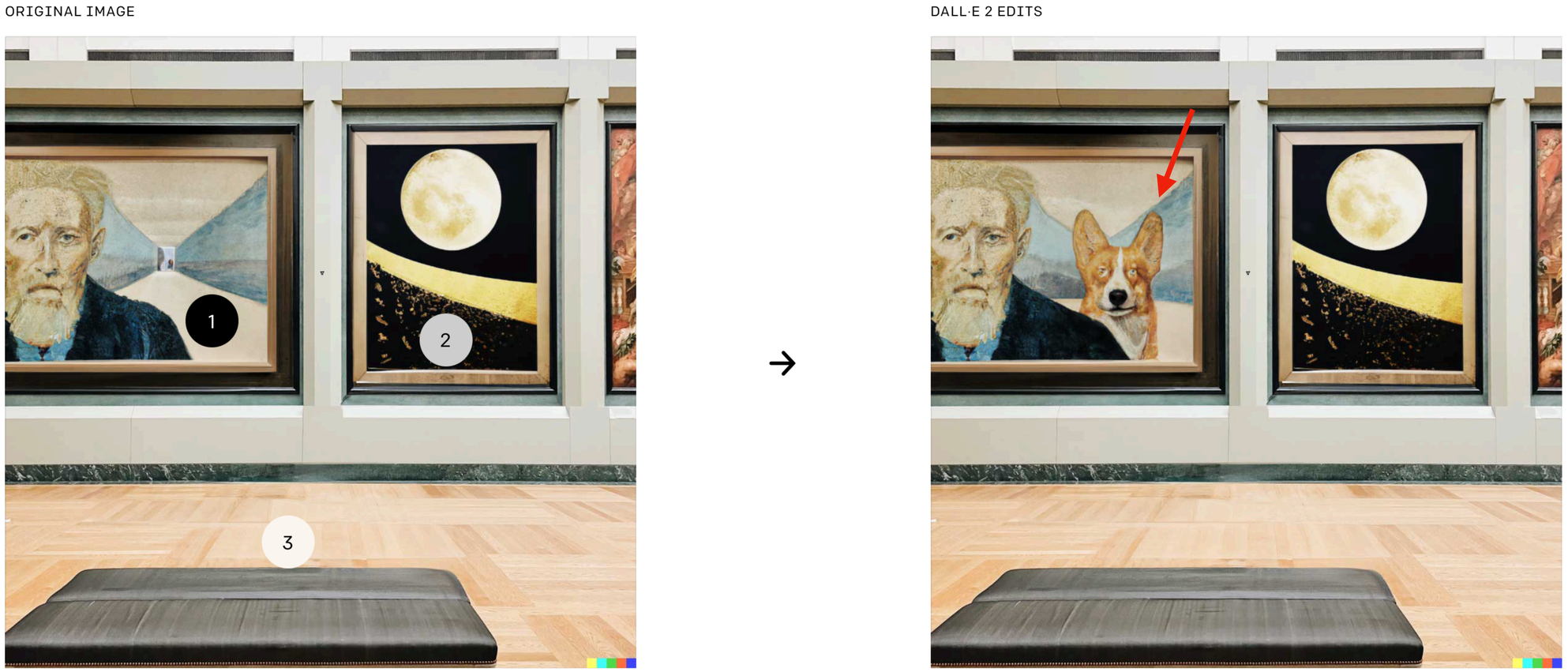


Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]

Background

Generative models for content creation

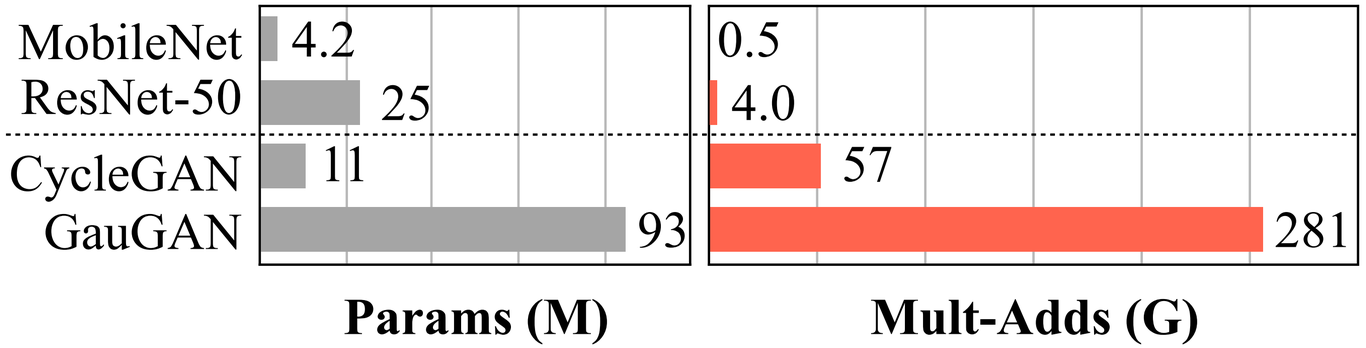
- Diffusion models for image editing



Hierarchical Text-Conditional Image Generation with CLIP Latents [Ramesh et al., 2022]

Difficulty: GANs are Computational-intensive

Generative models are more expensive than recognition models



Hierarchical Text-Conditional Image Generation with CLIP Latents [Ramesh et al., 2022]

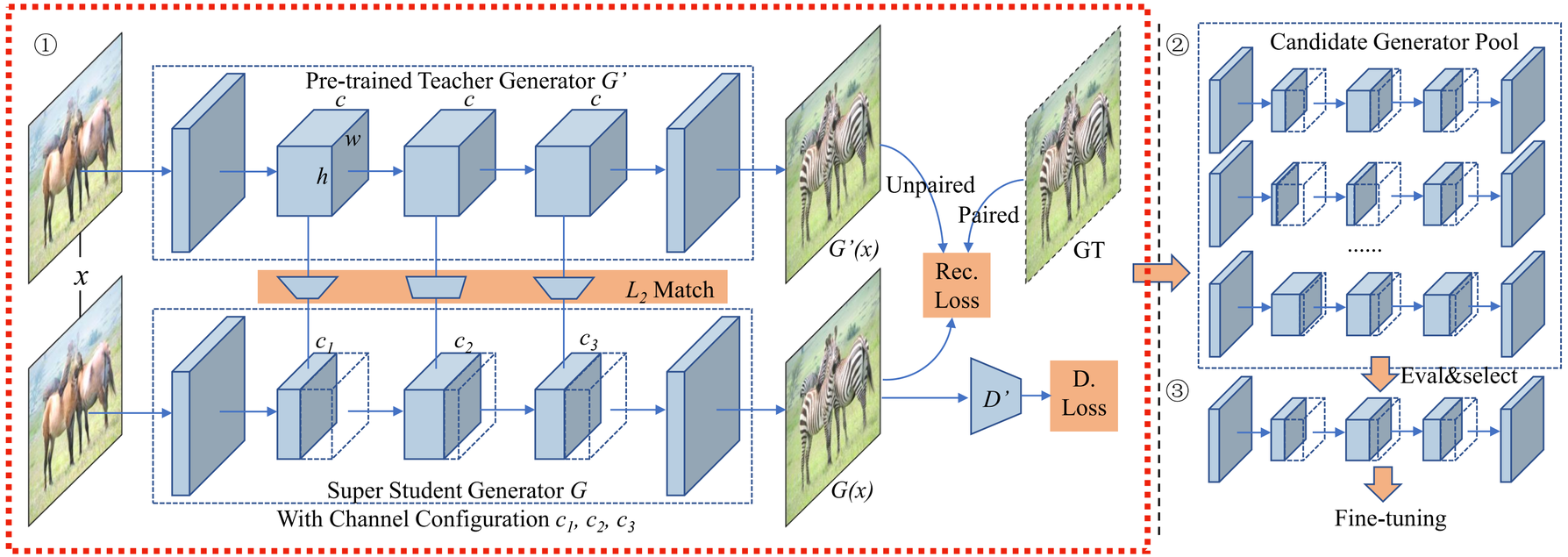
Lecture Plan

Efficient generative models

1. **GAN Compression (compress generators with NAS+distillation)**
2. Anycost GAN (dynamic cost vs. quality trade-off)
3. Spatially Sparse Inference (save computation by only updating edited regions)
4. Differentiable Augmentation (data efficient training of GANs)

GAN Compression

Compressing conditional GANs



Reconstruction Loss

$$\mathcal{L}_{\text{recon}} = \begin{cases} \|G(x) - y\| & \text{paired cGANs} \\ \|G(x) - G'(x)\| & \text{unpaired cGANs} \end{cases}$$

Distillation Loss

$$\mathcal{L}_{\text{distill}} = \sum_{k=1}^{n} \|G_k(x) - f_k(G'_k(x))\|$$

cGAN Loss

$$\mathcal{L}_{\text{cGAN}} = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_x[\log(1 - D(x, G(x)))]$$

Training Objective

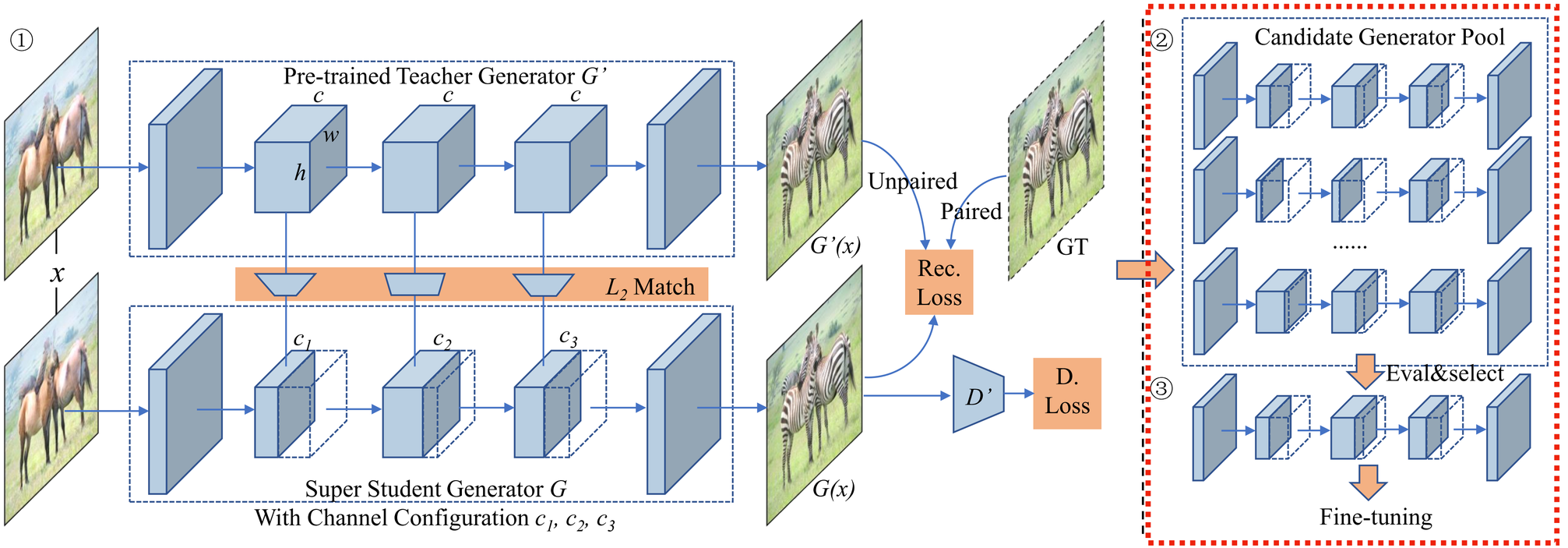
$$\mathcal{L}(x) = \mathcal{L}_{\text{cGAN}}(x) + \lambda_{\text{recon}}\mathcal{L}_{\text{recon}}(x) + \lambda_{\text{distill}}\mathcal{L}_{\text{distill}}(x)$$

GAN Compression: Learning Efficient Architectures for Conditional GANs [Li et al., 2020]

GAN Compression

Automated model surgery

- Neural architecture search: automated channel reduction



GAN Compression: Learning Efficient Architectures for Conditional GANs [Li et al., 2020]

GAN Compression

Results: compressing Pix2pix models

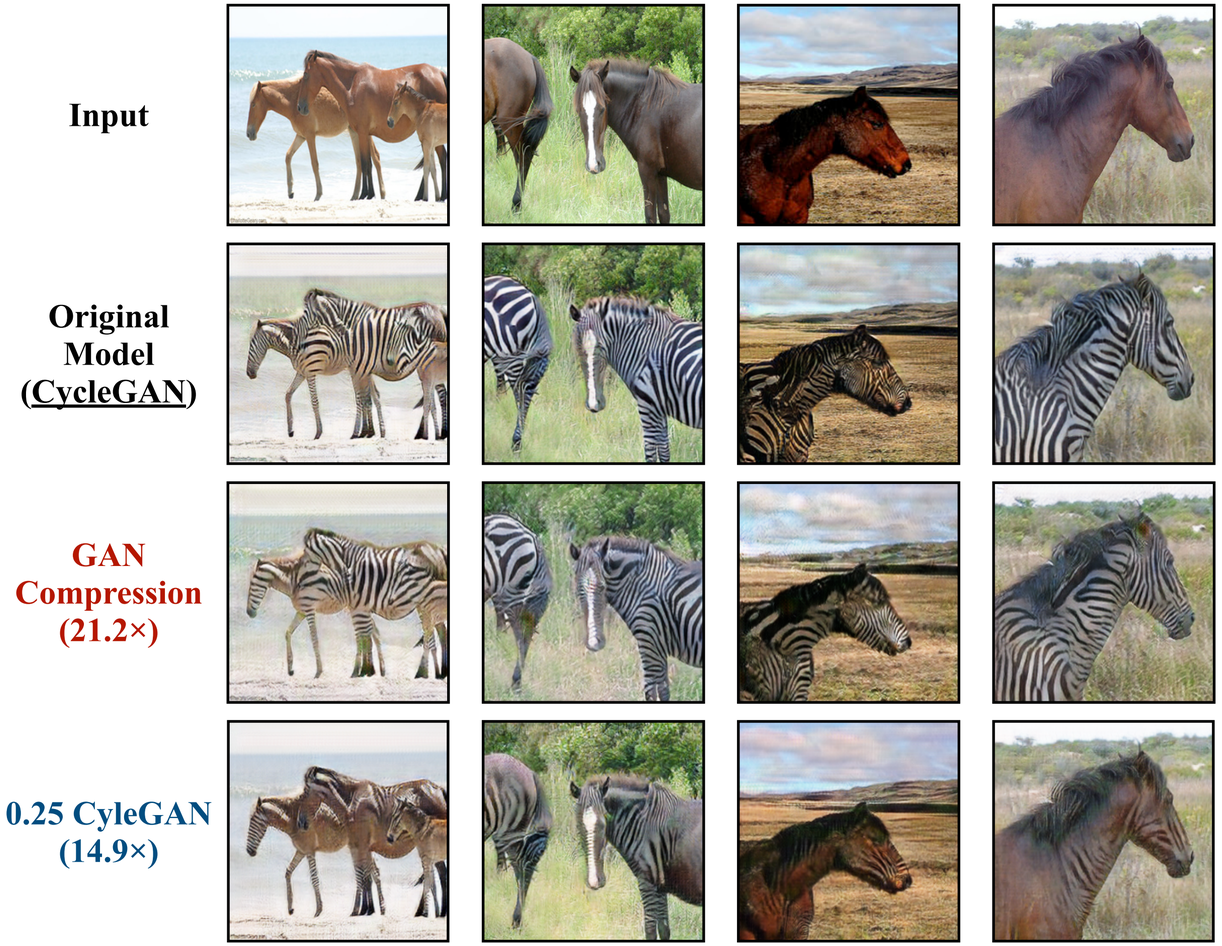


Image-to-Image Translation with Conditional Adversarial Networks [Isola et al., 2016]

GAN Compression

Results: compressing GauGAN

Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]

GAN Compression

Accelerating Horse2Zebra

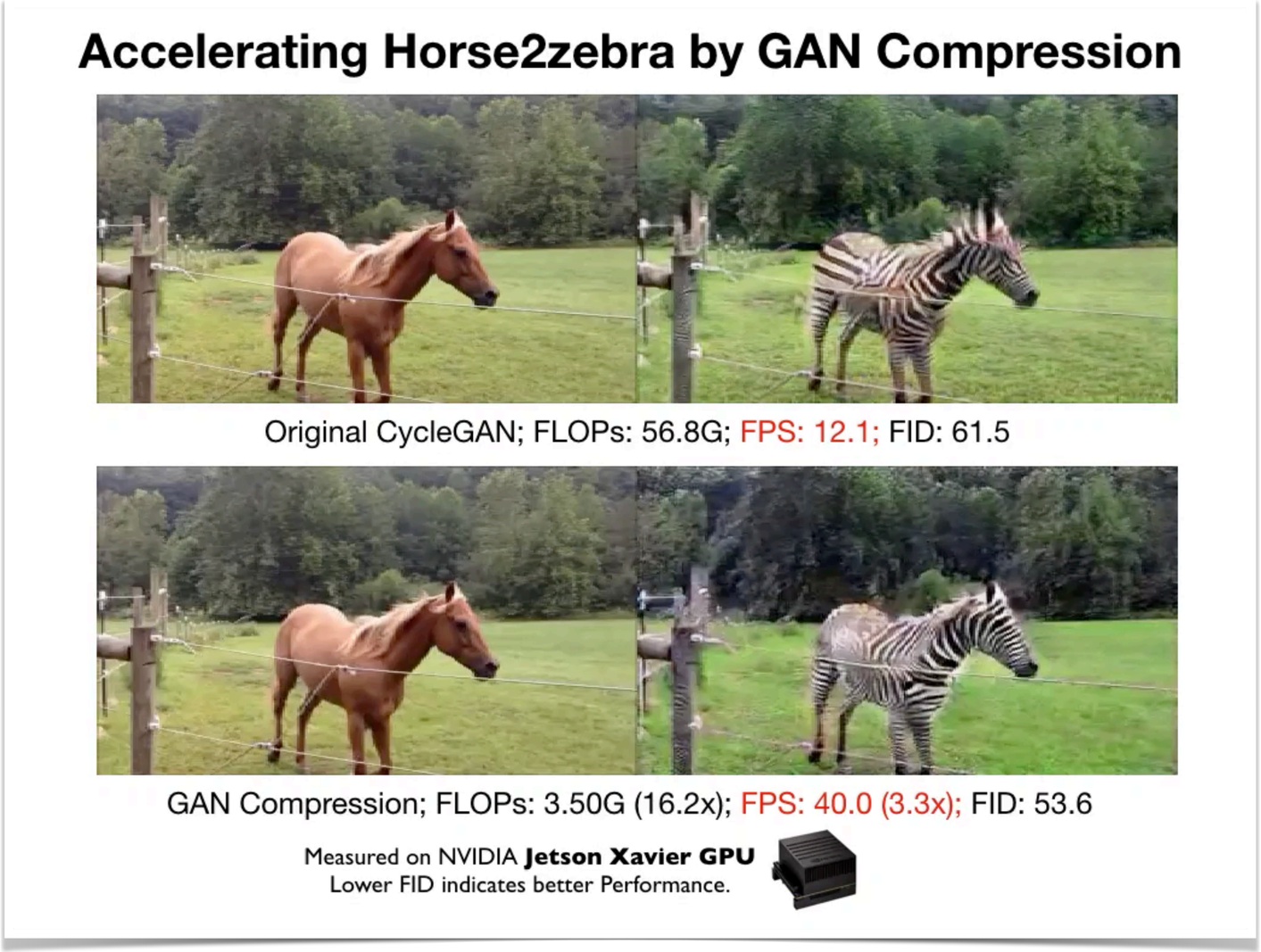


Image-to-Image Translation with Conditional Adversarial Networks [Isola et al., 2016]

Summary of Today's Lecture

In this lecture, we introduce:

1. Efficient video understanding
    1. 2D CNNs for video understanding
    2. 3D CNNs for video understanding
    3. Temporal Shift Module (TSM)
    4. Other efficient methods for video understanding
2. Efficient generative models
    1. GAN Compression (compress generators with NAS+distillation)

In the next lecture, we will introduce:

2. Anycost GAN (dynamic cost vs. quality trade-off)
3. Spatially Sparse Inference (save computation by only updating edited regions)
4. Differentiable Augmentation (data efficient training of GANs)

Lecture Plan

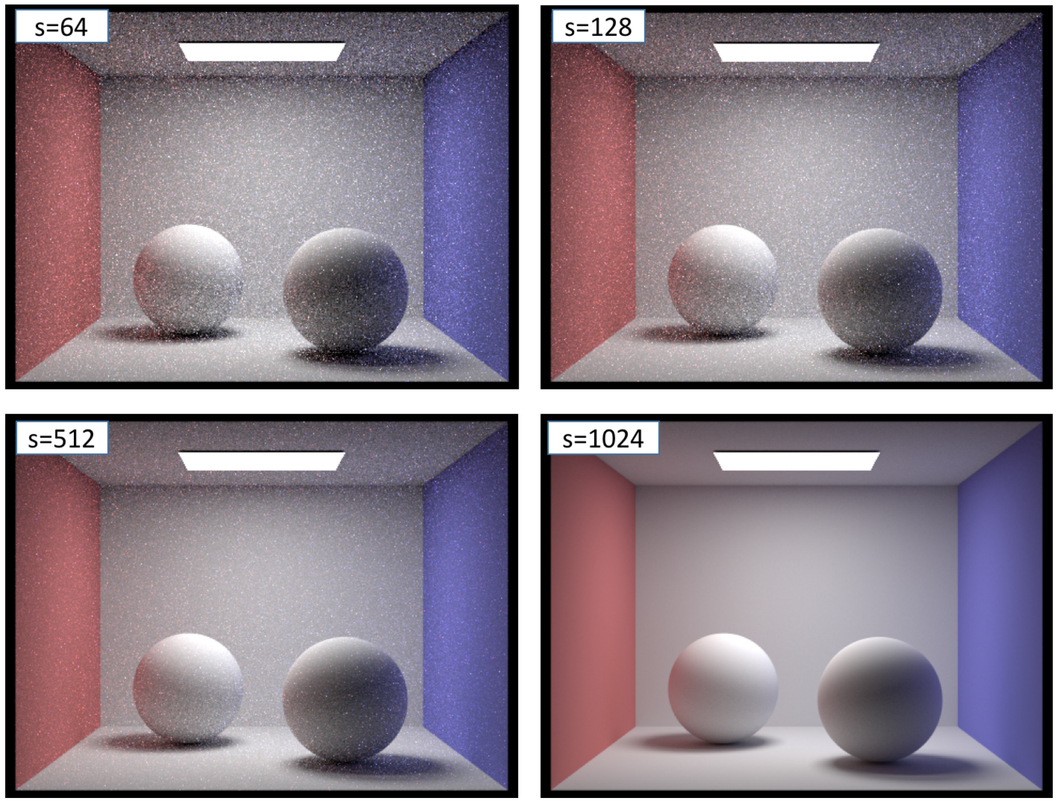
Efficient generative models

1. GAN Compression (compress generators with NAS+distillation)
2. **Anycost GAN (dynamic cost vs. quality trade-off)**
3. Spatially Sparse Inference (save computation by only updating edited regions)
4. Differentiable Augmentation (data efficient training of GANs)

Anycost GANs

Motivation

- In the ray-tracing pipeline, people get fast previews by reducing the number of sampled rays
- Can we do the same for GANs?

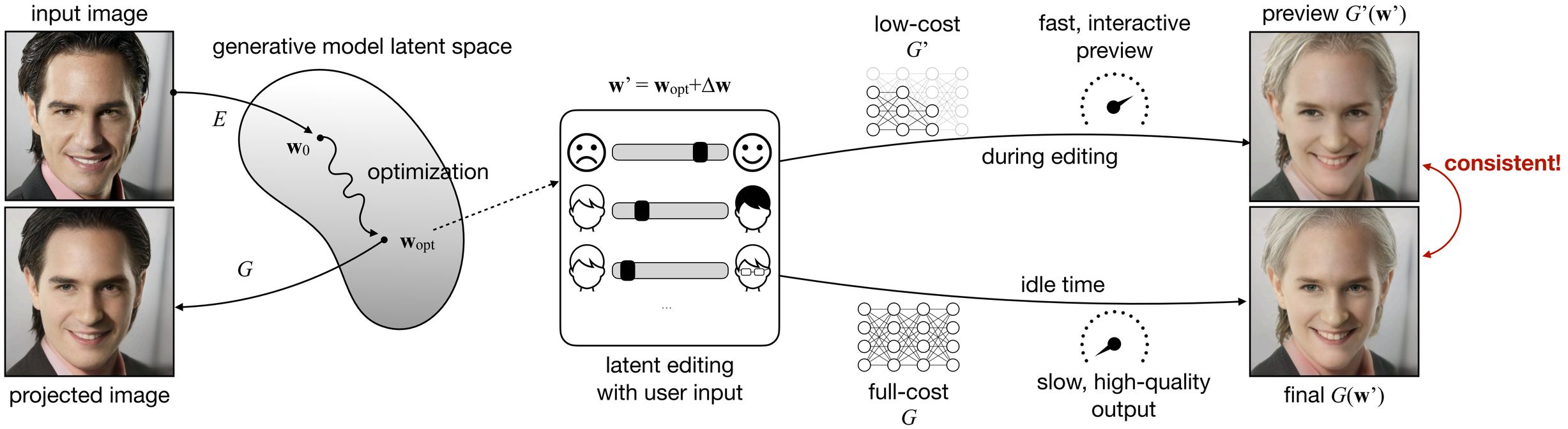


https://ceciliavision.github.io/graphics/a3/

Anycost GANs

Motivation

- In the ray-tracing pipeline, people get fast previews by reducing the number of sampled rays
- Can we do the same for GANs?



Anycost gans for interactive image synthesis and editing [Lin et al., 2021]

Anycost GANs

Learning anycost GANs

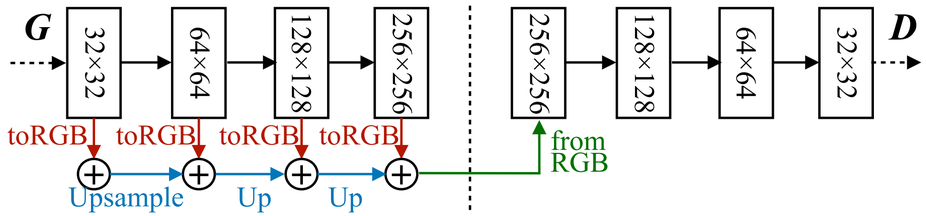
- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

Anycost GANs

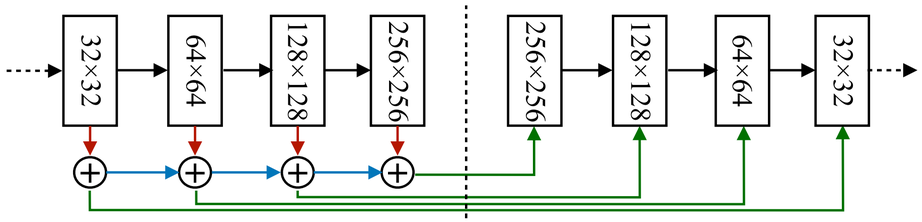
Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

- **StyleGAN2**: feed only highest resolution



- **MSG-GAN**: feed all resolutions

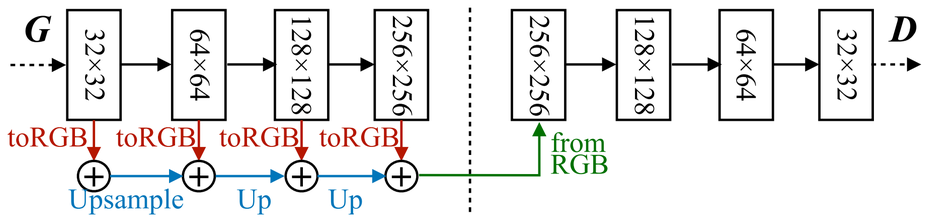


Anycost GANs

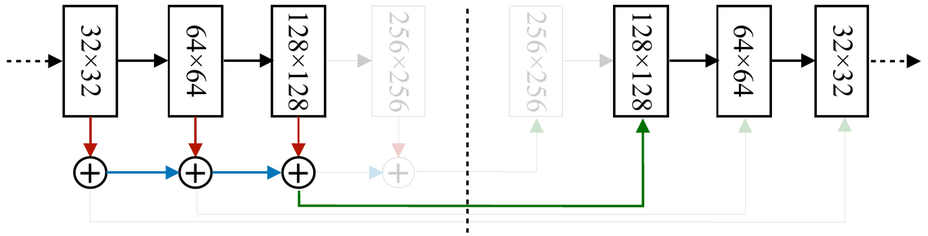
Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

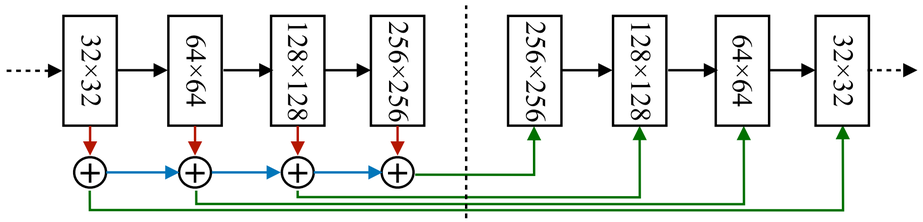
- **StyleGAN2**: feed only highest resolution



- **Ours**: sampling-based multi-res



- **MSG-GAN**: feed all resolutions

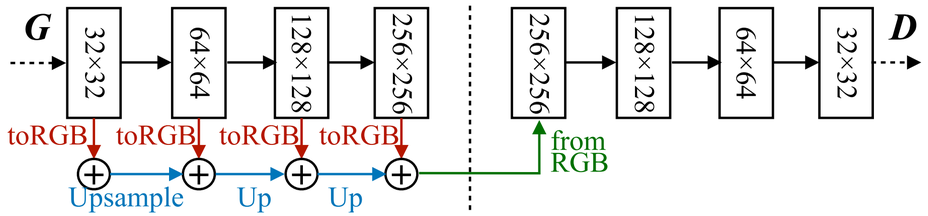


Anycost GANs

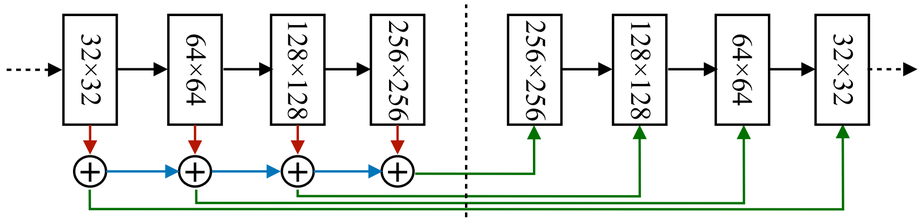
Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

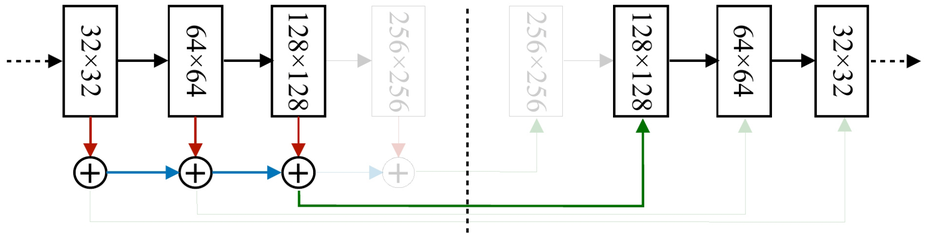
- **StyleGAN2**: feed only highest resolution



- **MSG-GAN**: feed all resolutions



- **Ours**: sampling-based multi-res



better FID↓ compared to single-resolution models

Resolution	1024	512	256	128	64	32
Single-resolution	3.25	4.17	3.76	4.04	3.32	2.41
MSG-GAN [41]	-	-	4.79	6.34	2.7	3.04
Ours (low res)	-	-	3.49	**3.26**	**2.52**	**2.18**
Ours (high res)	**2.99**	**3.08**	**3.35**	3.98	-	-

Anycost GANs

Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

StyleGAN2

32x32 64x64 128x128 256x256

Ours

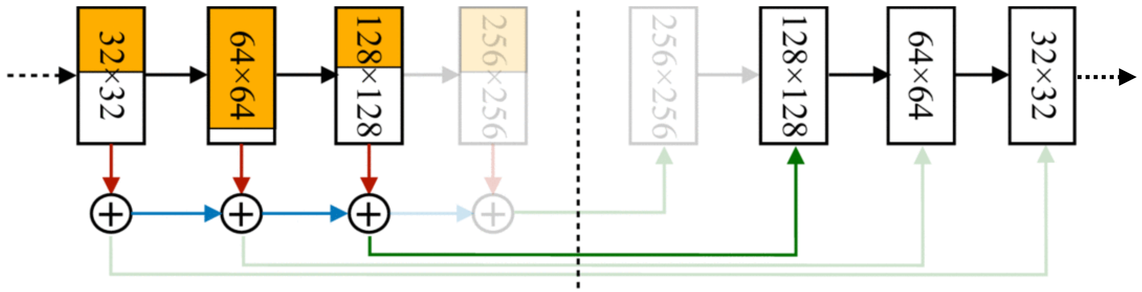
32x32 64x64 128x128 256x256

Anycost GANs

Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

    - Sample adaptive channel numbers during training

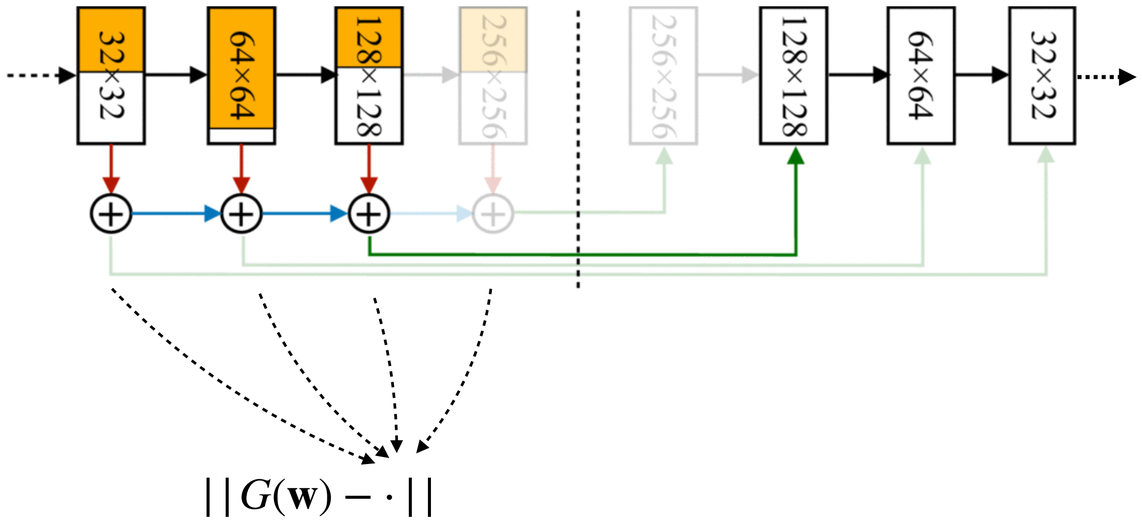


Anycost GANs

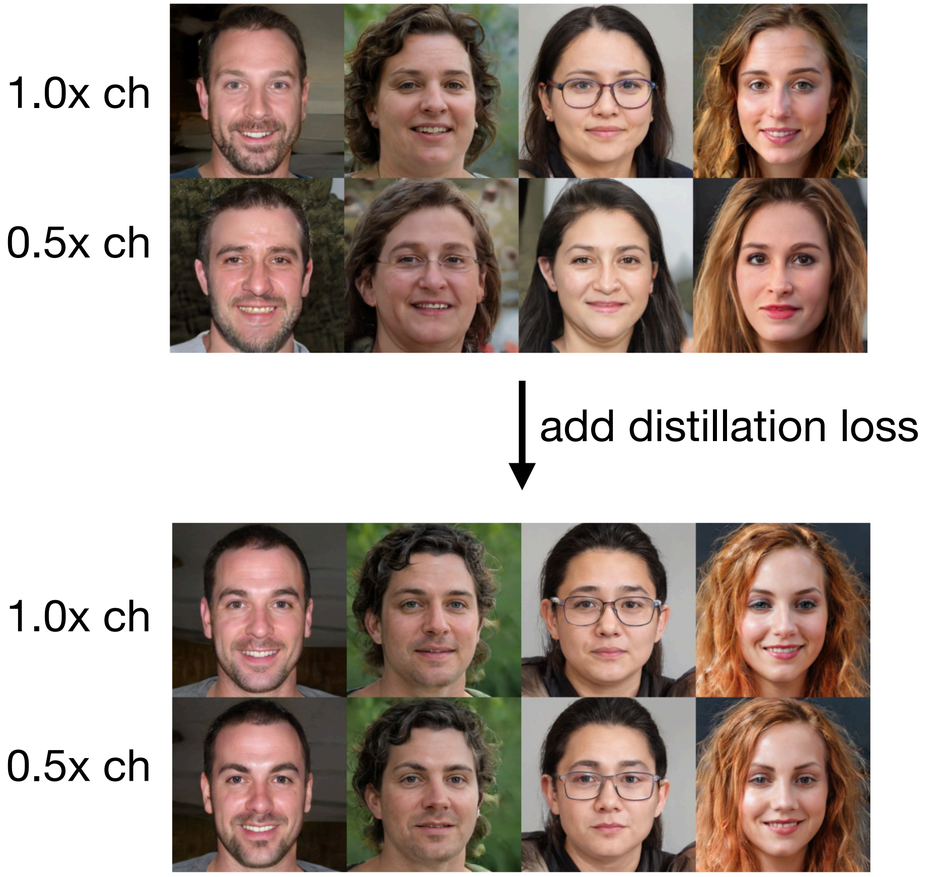
Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

- Sample adaptive channel numbers during training



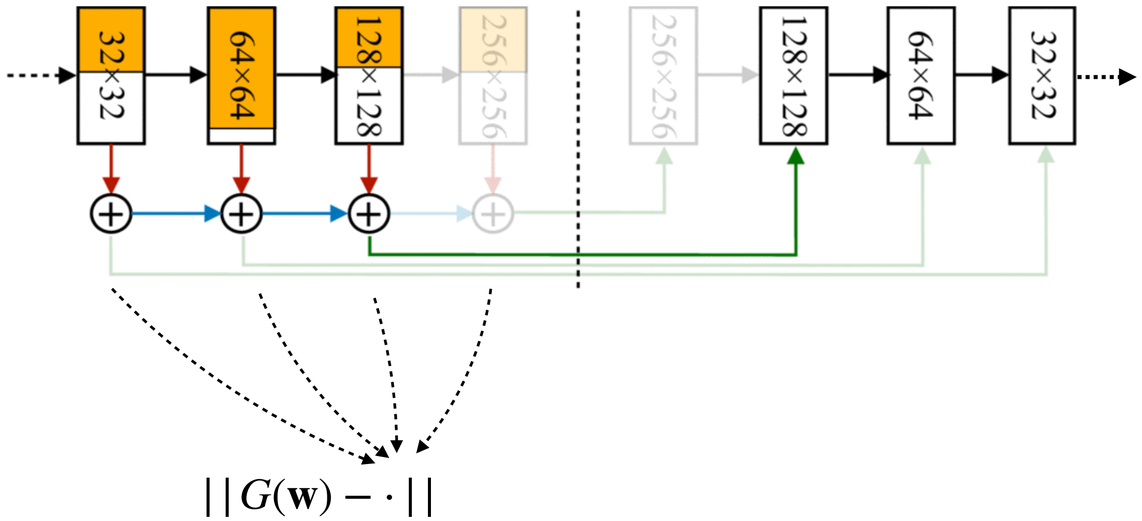
Problem 1: inconsistency between subnets



Anycost GANs

Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers
    - Sample adaptive channel numbers during training



Problem 2: bad FID for some subnets

FID-70k↓	resolution 256				resolution 128			
	1×	0.75×	0.5×	0.25×	1.0×	0.75×	0.5×	0.25×
vanilla	3.80	4.64	6.20	10.39	4.04	4.99	5.78	11.15
same D	3.63	3.91	5.41	14.01	7.25	6.81	5.92	7.57

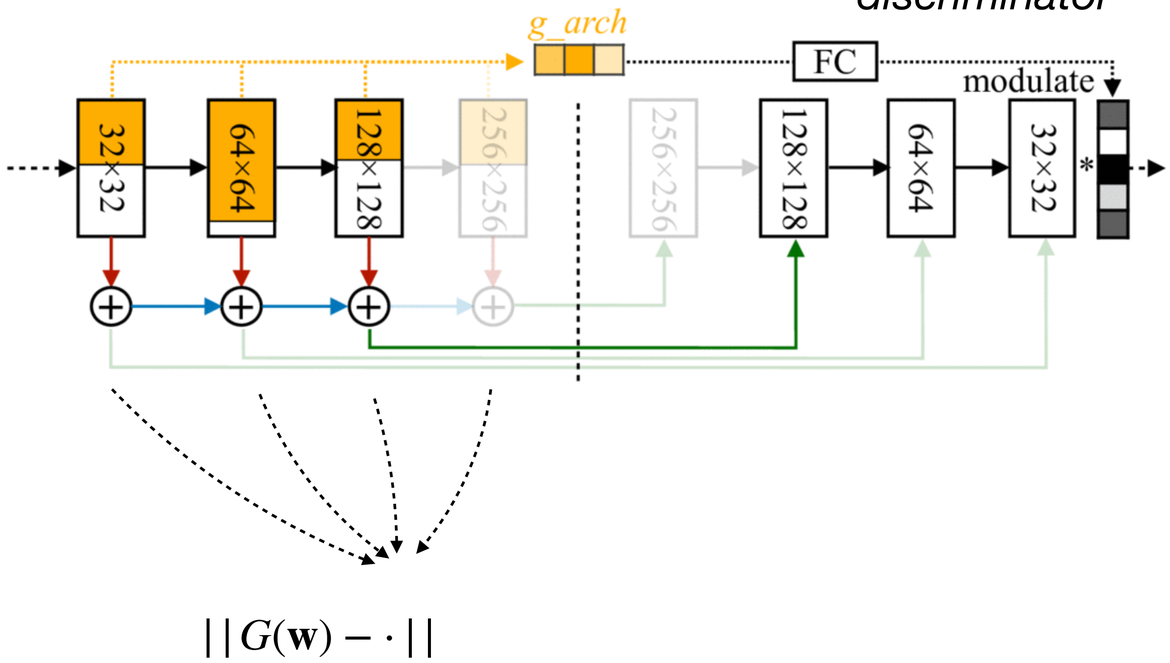
A **single** discriminator cannot provide effective feedback for **different** sub-generators

Anycost GANs

Learning anycost GANs

- We train the model to produce consistent output at
  - Different resolutions
  - Different channel numbers

G-conditioned discriminator



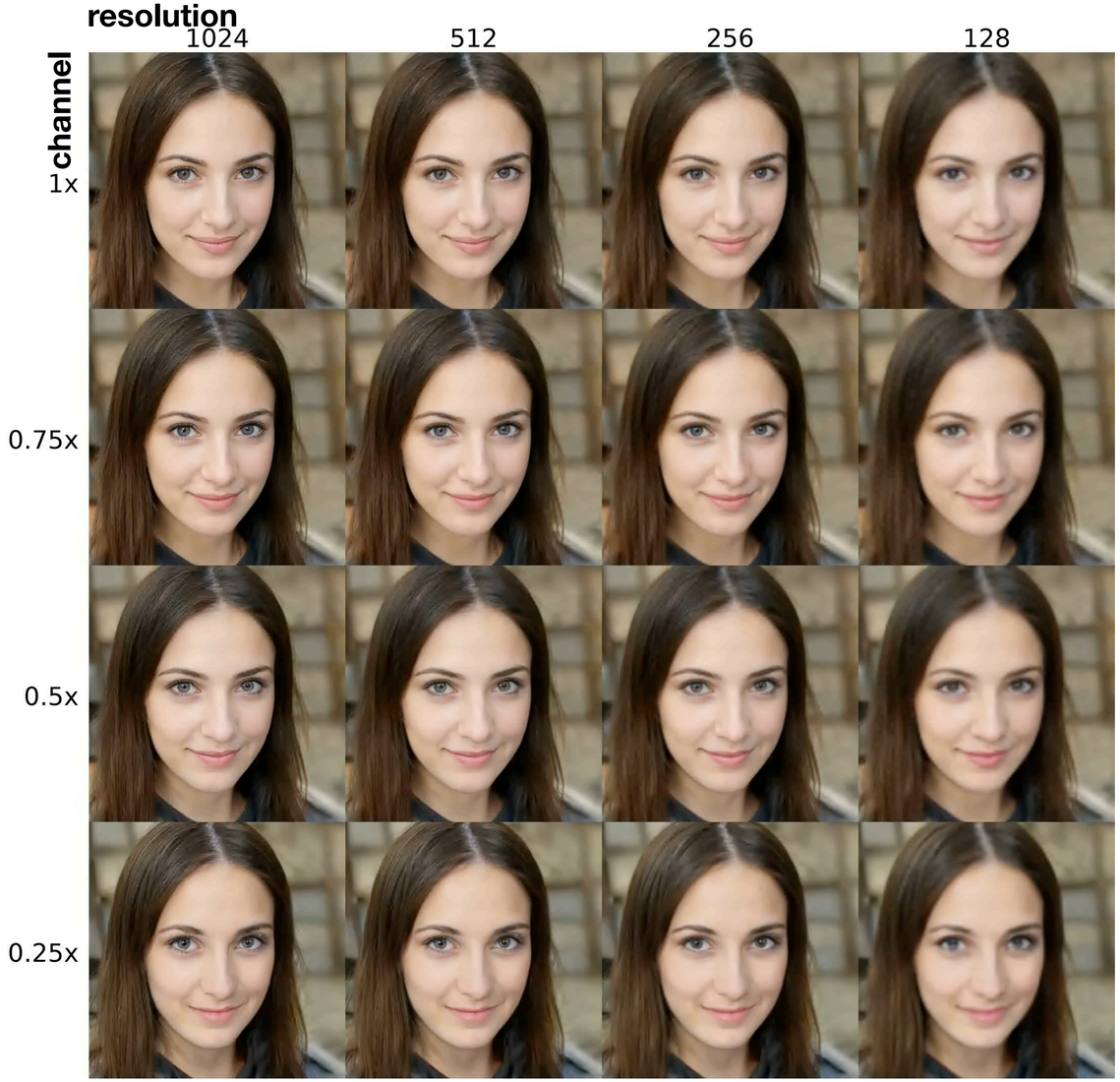
Problem 2: bad FID for some subnets

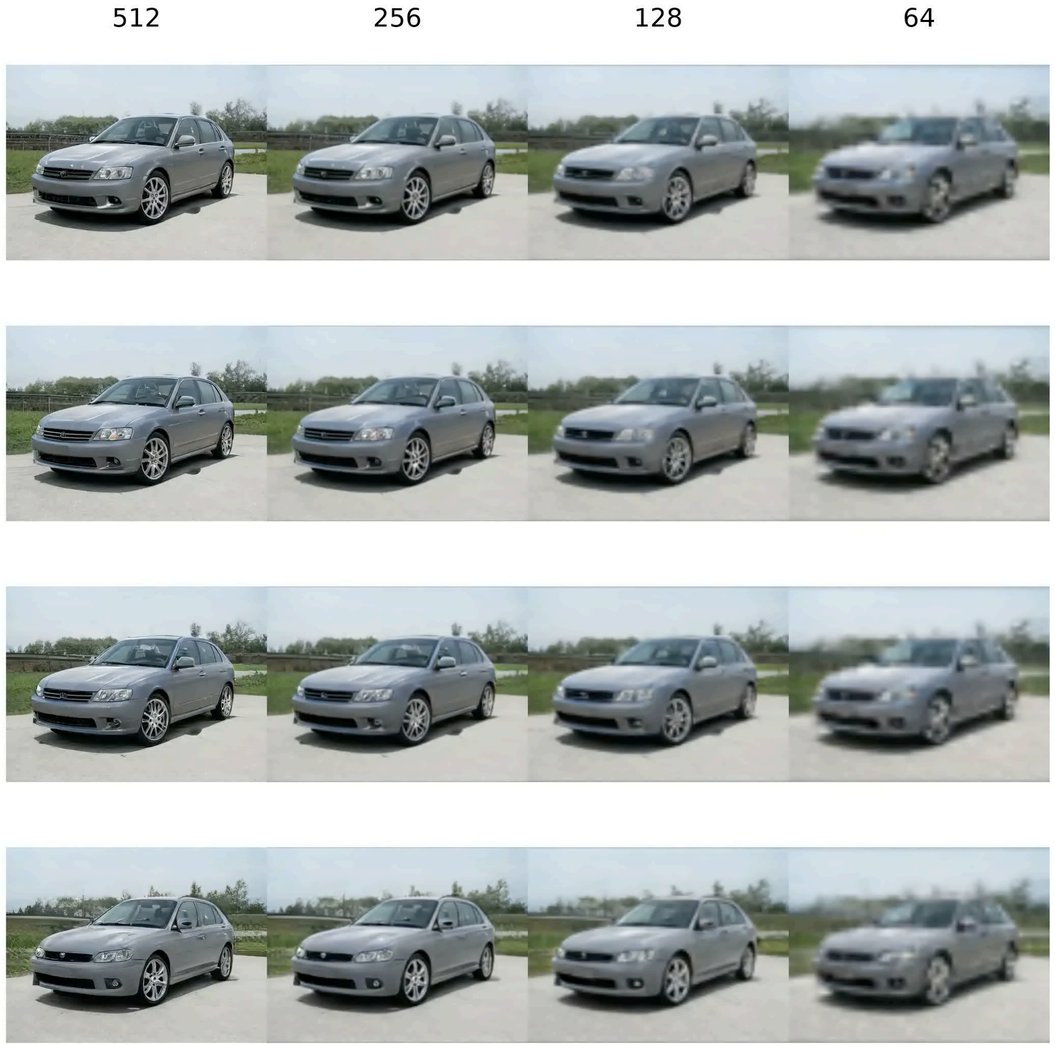
FID-70k↓	resolution 256				resolution 128			
	1×	0.75×	0.5×	0.25×	1.0×	0.75×	0.5×	0.25×
vanilla	3.80	4.64	6.20	10.39	4.04	4.99	5.78	11.15
same D	3.63	3.91	5.41	14.01	7.25	6.81	5.92	7.57
conditioned (Ours)	3.73	3.86	4.64	8.06	3.30	3.28	3.69	5.55

With G-conditioned discriminator, we can achieve consistent improvement.

Anycost GANs

Forward Consistency at different resolutions/channels





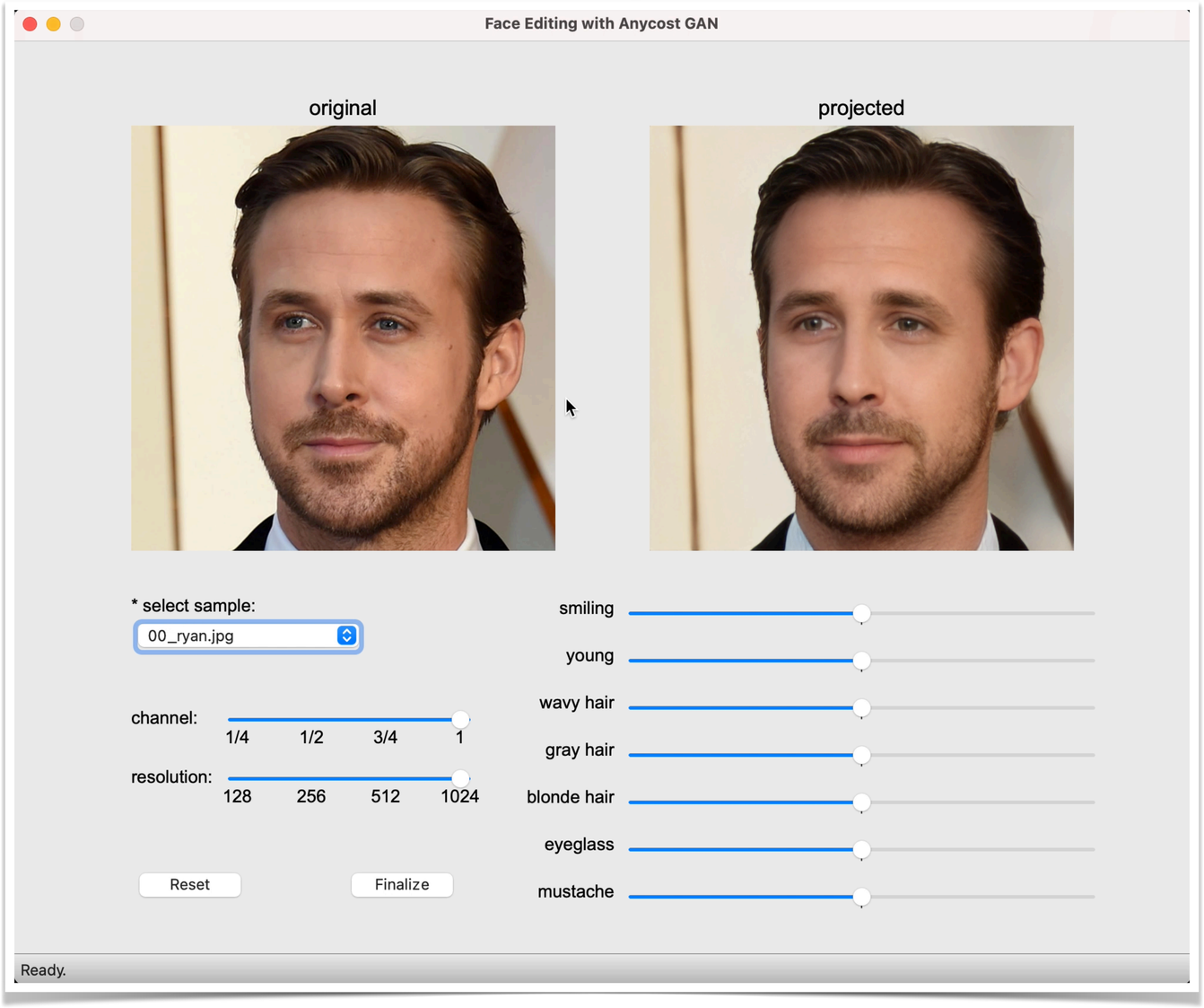
Anycost GANs

Forward Consistency at different resolutions/channels

Fine-grained computational budgets

Anycost GANs

Demo



Lecture Plan

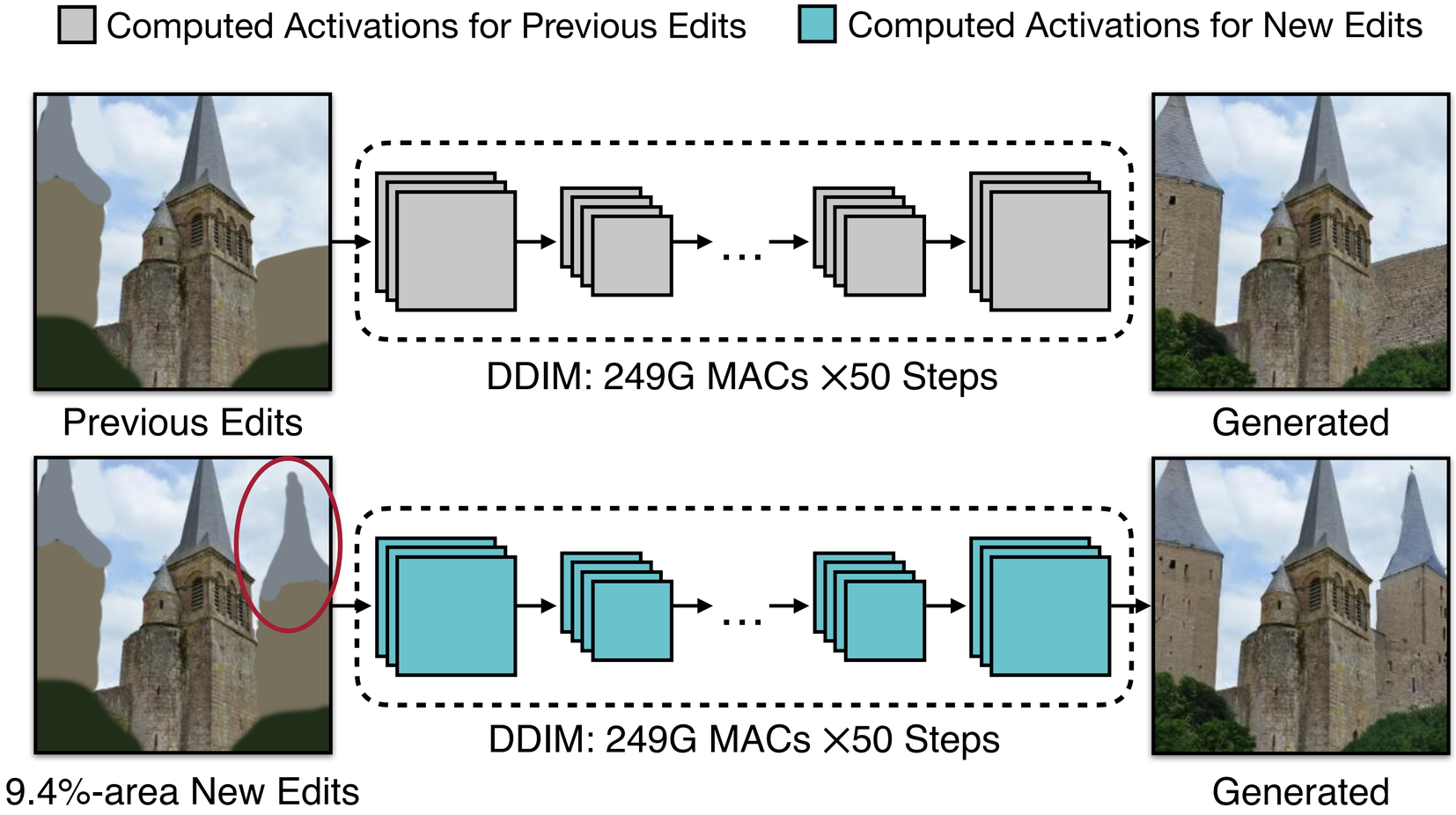
Efficient generative models

1. GAN Compression (compress generators with NAS+distillation)
2. Anycost GAN (dynamic cost vs. quality trade-off)
3. **Spatially Sparse Inference (save computation by only updating edited regions)**
4. Differentiable Augmentation (data efficient training of GANs)

Spatially Sparse Inference

Exploring the spatial redundancy

- Traditionally, we need to regenerate the entire image even when a small region is edited

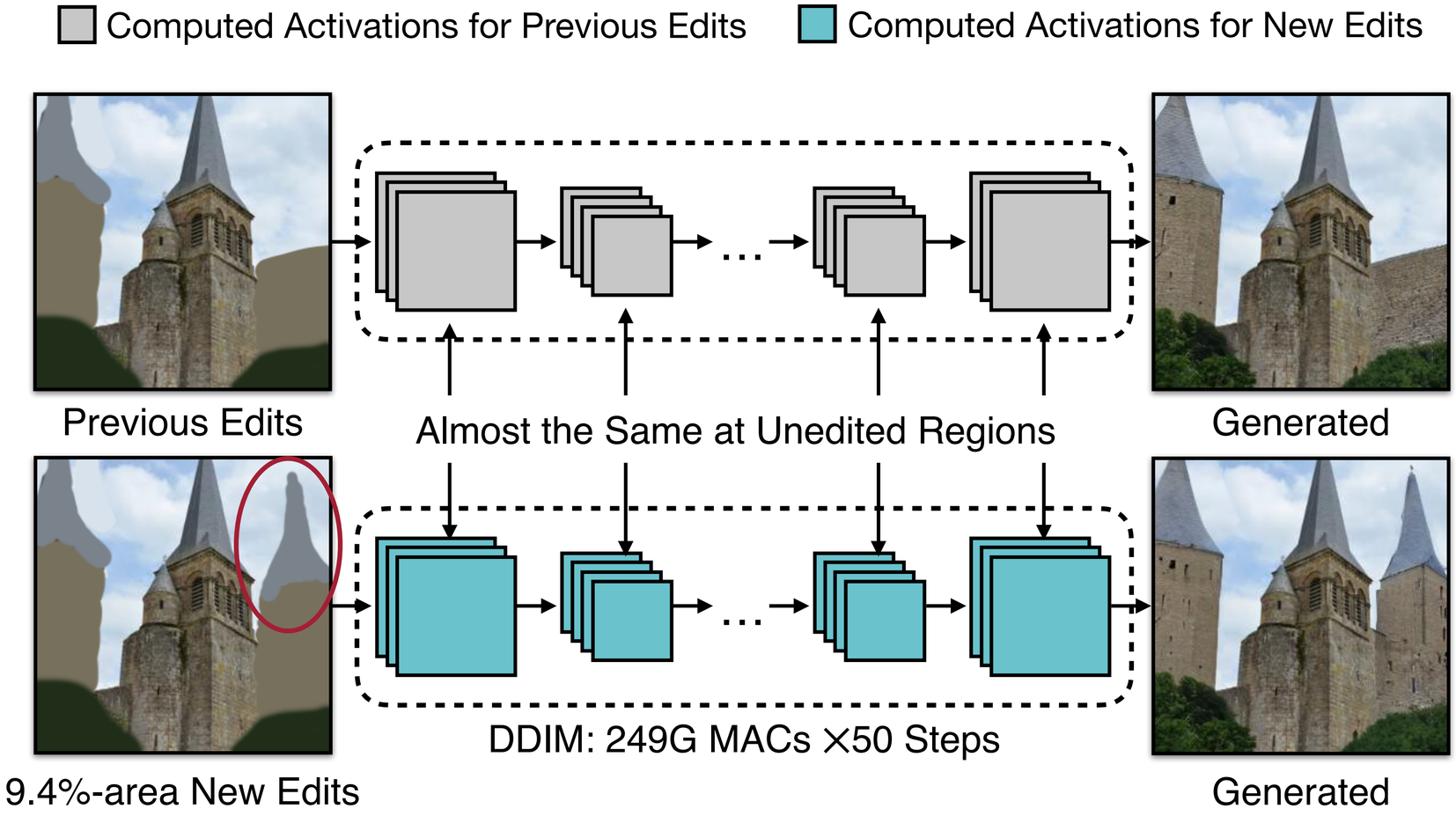


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Exploring the spatial redundancy

- Feature maps remain mostly the same in the unedited regions.

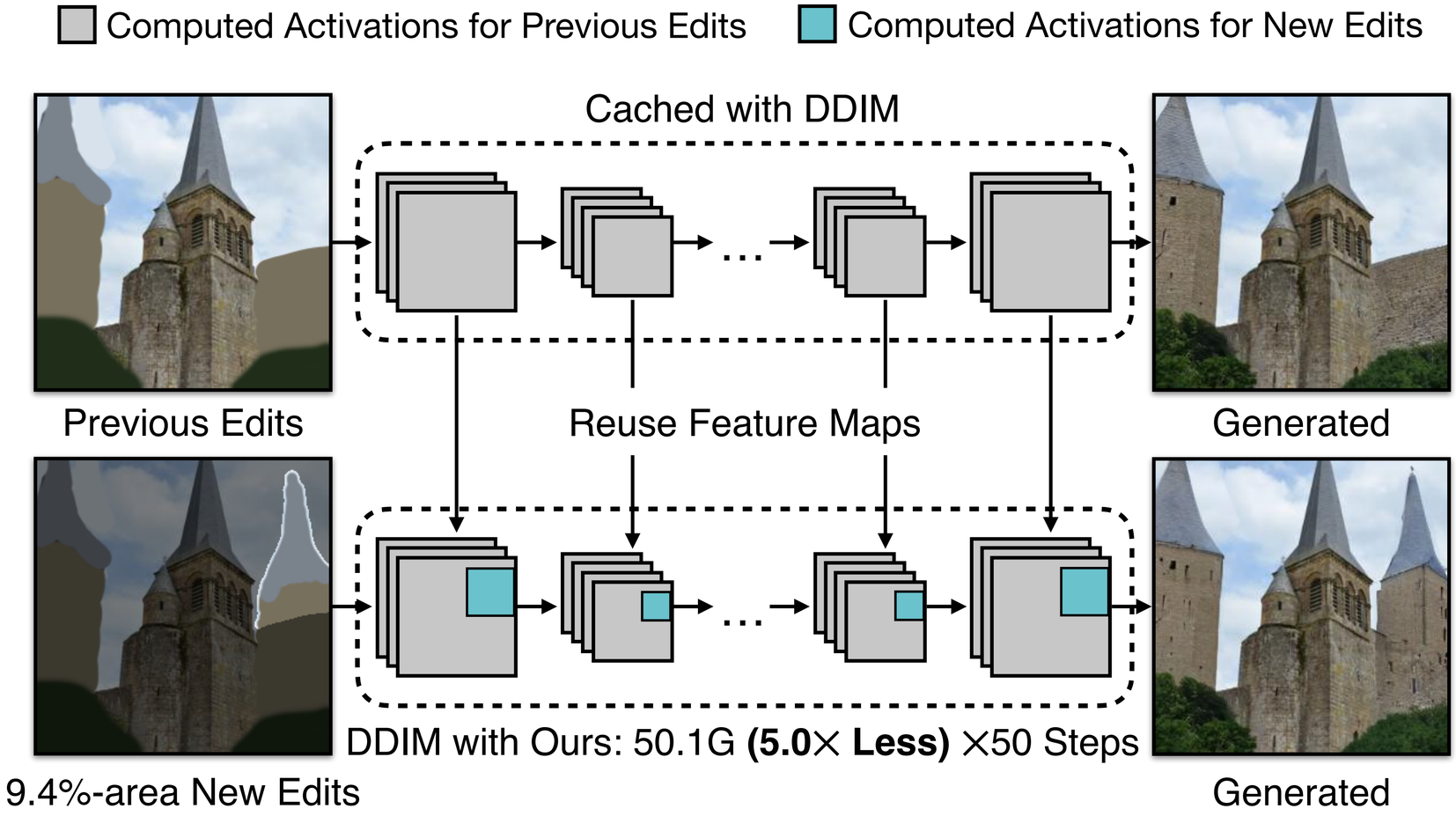


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Exploring the spatial redundancy

- Instead, we want to update only the edited region and **reuse** the cached activations for unedited

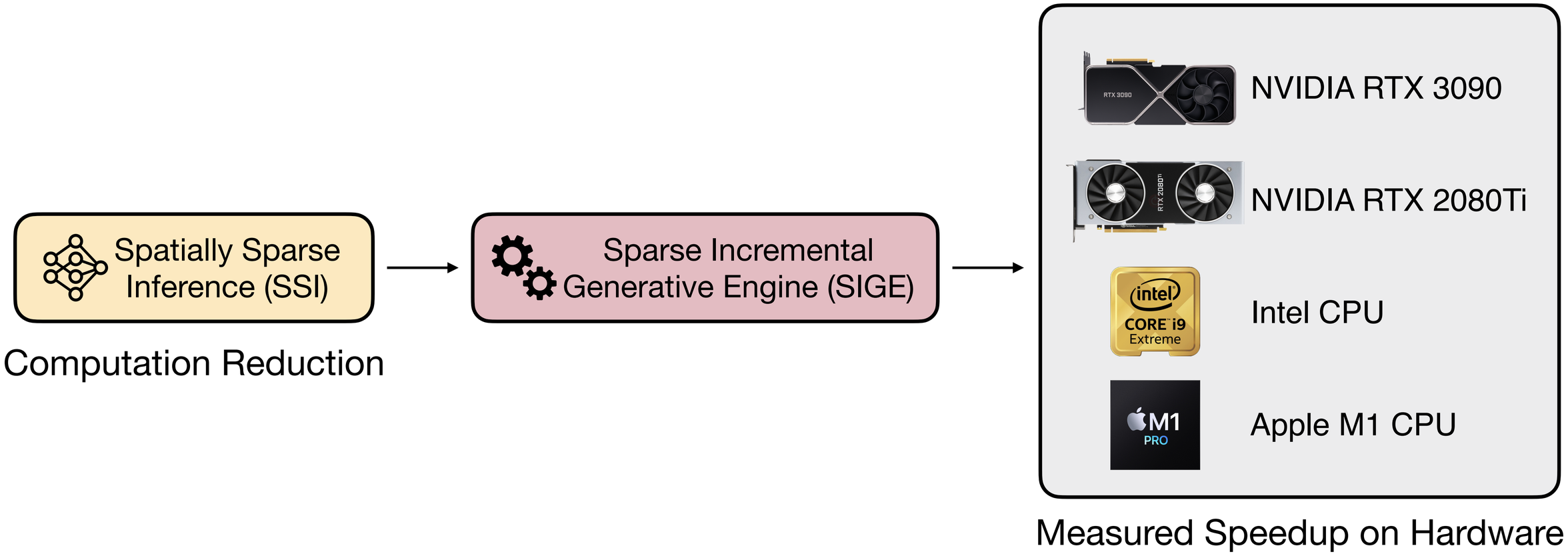


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Sparse Incremental Generative Engine (SIGE)

We need specialized system support to accelerate inference

- SIGE realizes the theoretical savings on various hardware settings.

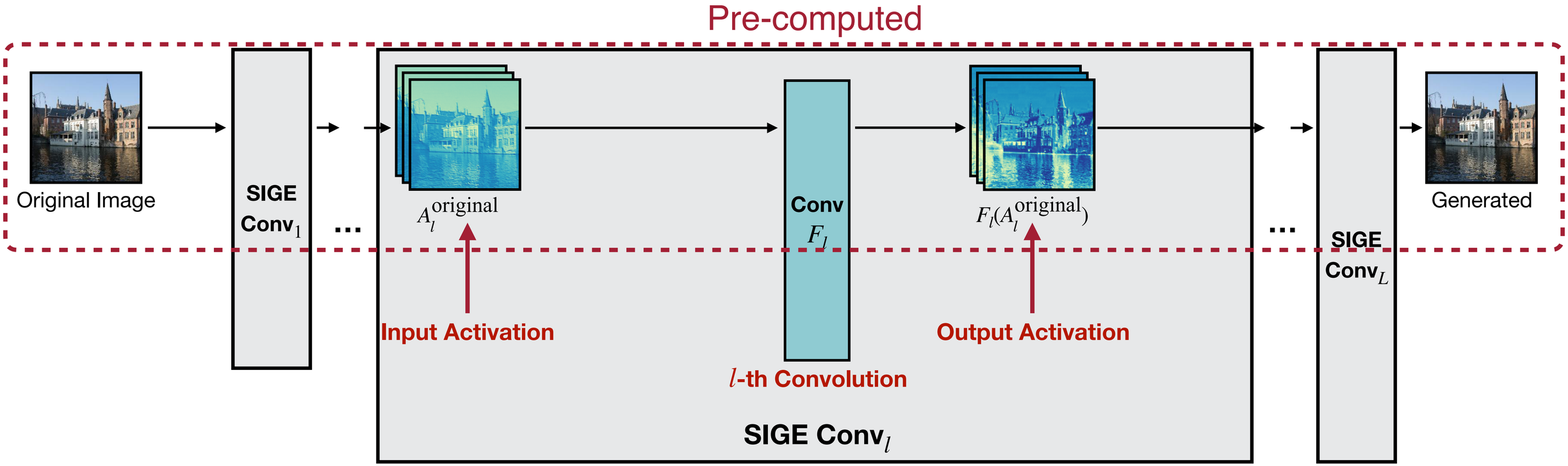


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Tiling-based Sparse Convolution

- The original image stream is pre-computed.

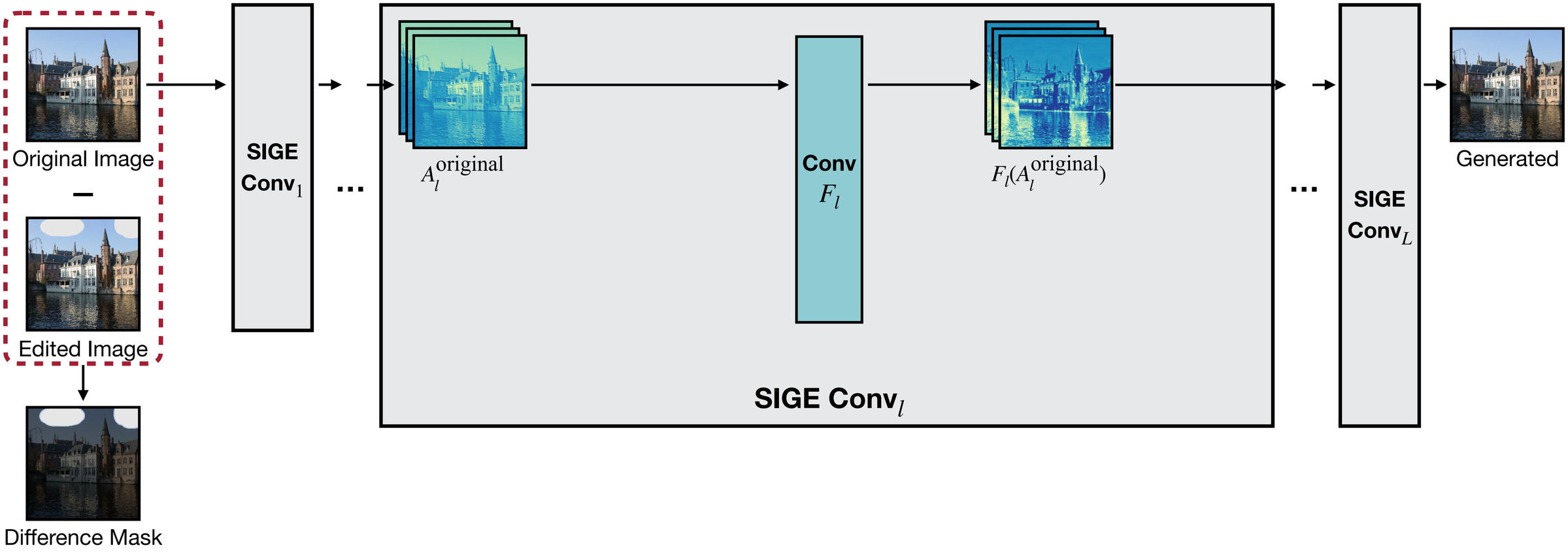


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Tiling-based Sparse Convolution

- Use a difference mask to locate the edited regions.

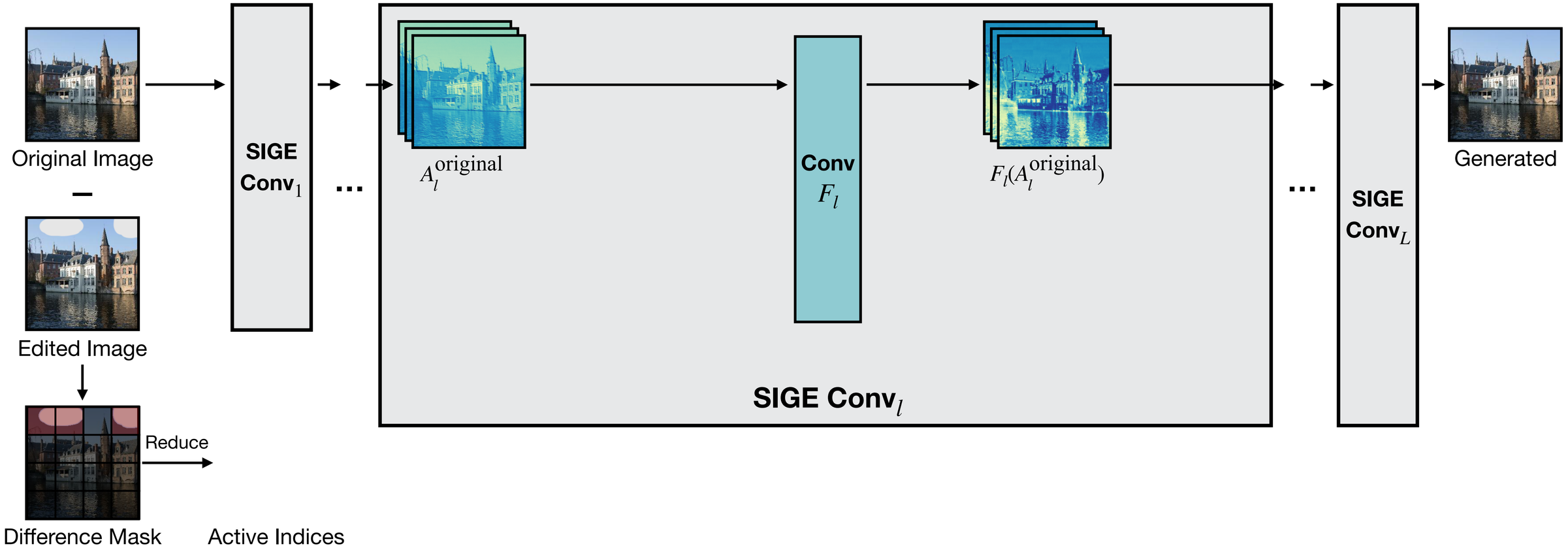


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Tiling-based Sparse Convolution

- Divide the difference mask into small blocks and reduce it to active indices.

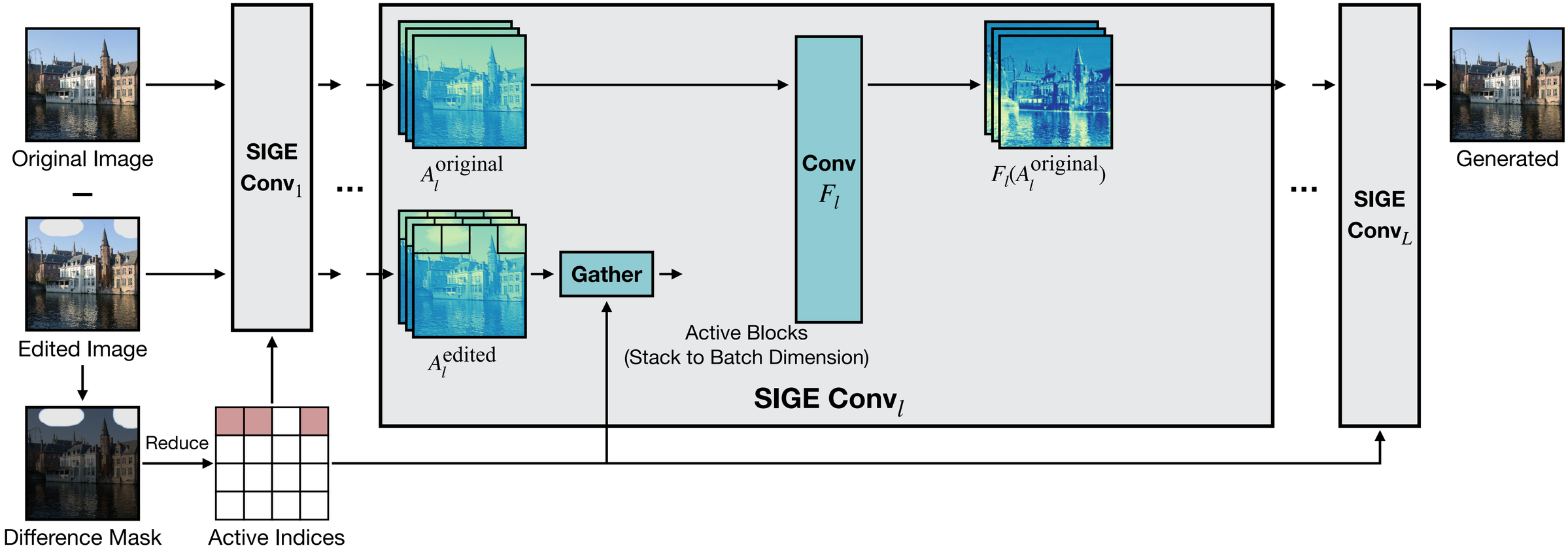


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Tiling-based Sparse Convolution

- Gather the active blocks along the batch dimension.

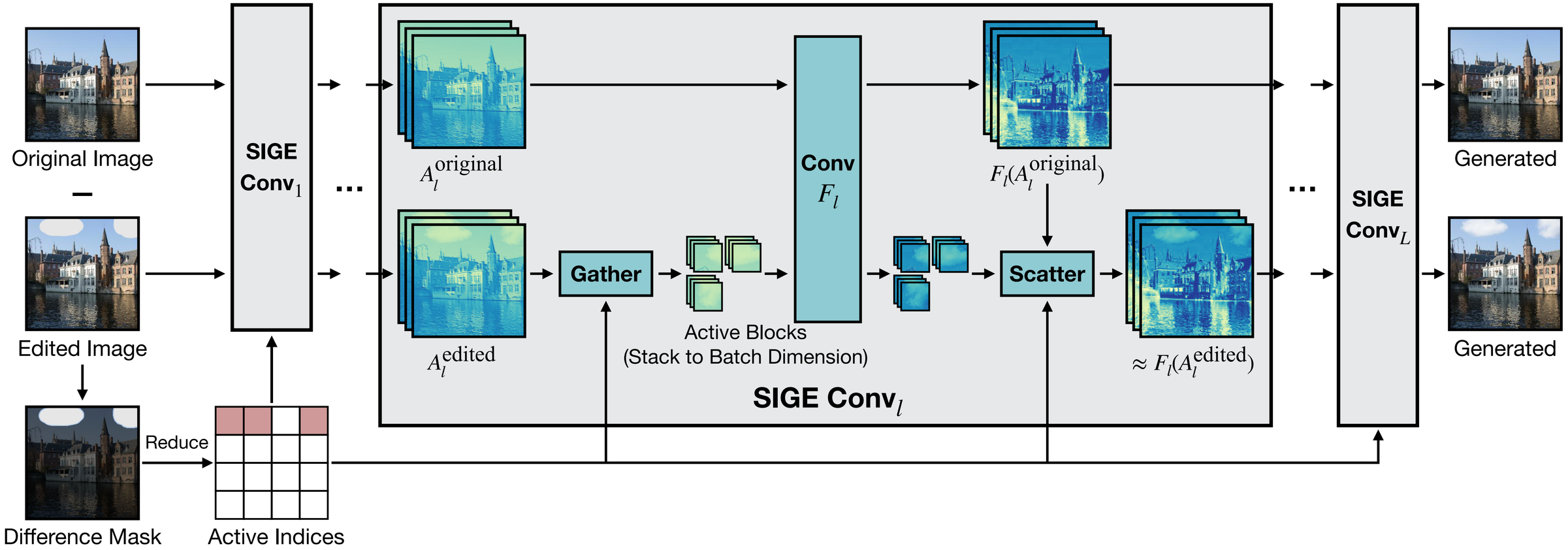


Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Tiling-based Sparse Convolution

- Scatter the output blocks into the original activation $F_l(A_l^{\text{original}})$.



Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]

Spatially Sparse Inference

Qualitative Results - DDIM

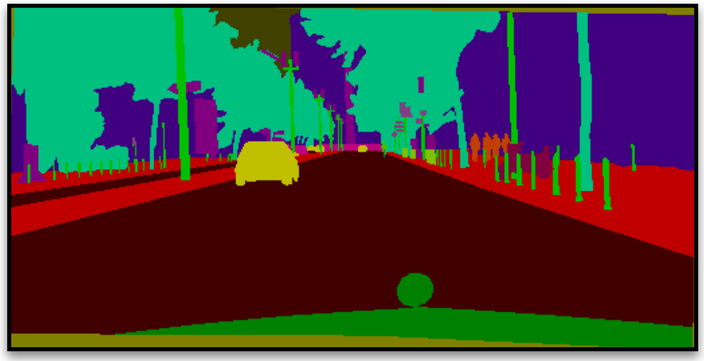
Original | 11.5% Edited | DDIM: 249G MACs | Ours: 61.3G (4.1×)

Original | 1.20% Edited | DDIM: 249G MACs | Ours: 33.4G (7.5×)

SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations [Meng et al., 2022]

Spatially Sparse Inference

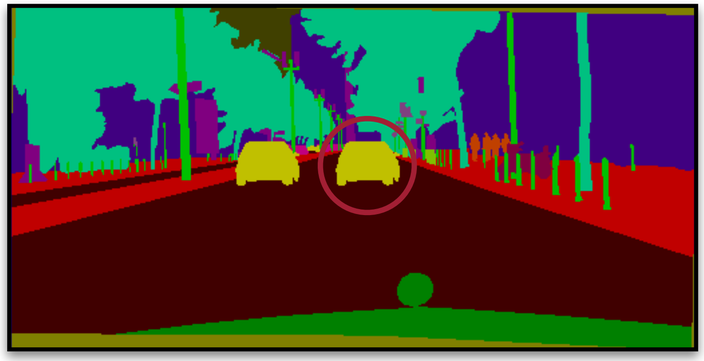
Quantitive Results - GauGAN (combined with GAN Compression)



Original

GauGAN: 281G MACs

GAN Compression: 31.2G (9.0×)



1.18% Edited

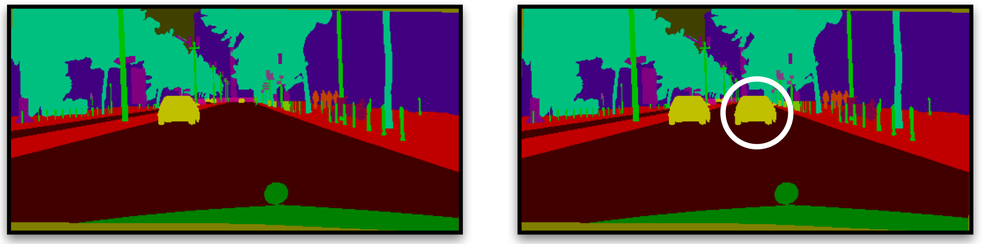
Ours: 15.3G (18×)

GAN Comp.+Ours: 5.59G (50×)

Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]

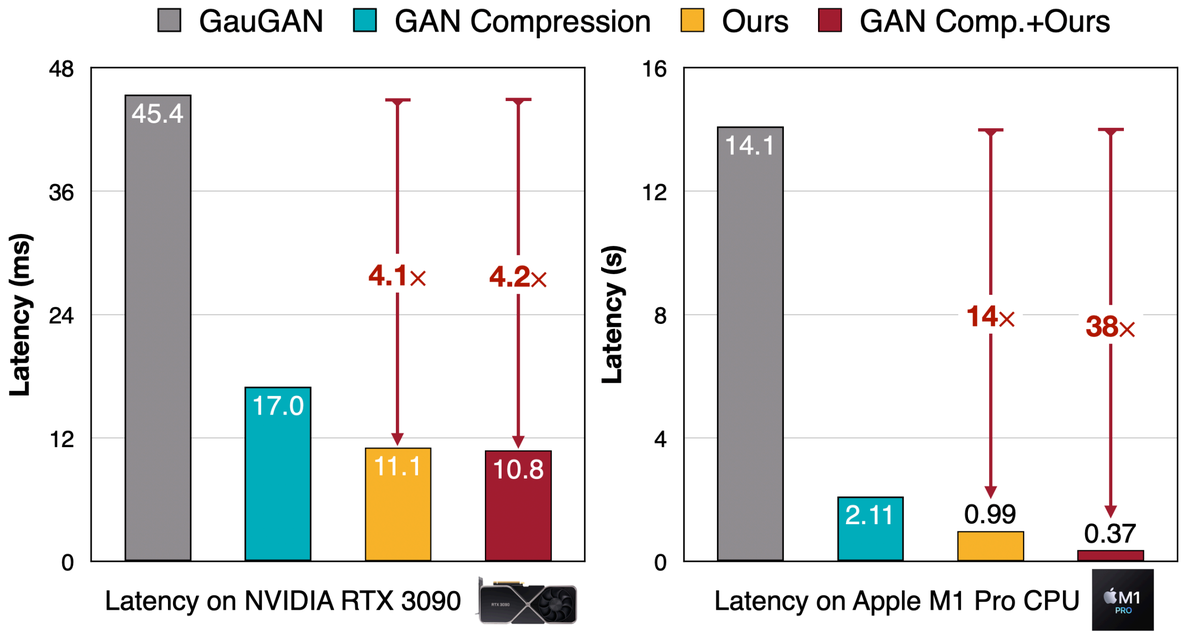
Spatially Sparse Inference

Qualitative Results - GauGAN (combined with GAN Compression)



Original

1.18% Edited



Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]

Lecture Plan

Efficient generative models

1. GAN Compression (compress generators with NAS+distillation)
2. Anycost GAN (dynamic cost vs. quality trade-off)
3. Spatially Sparse Inference (save computation by only updating edited regions)
4. **Differentiable Augmentation (data efficient training of GANs)**

Data-Efficient GANs

Big Data is Expensive

- It takes months or even years to collect the data, along with prohibitive annotation costs.

FFHQ dataset: **70,000** selective post-processed human faces

ImageNet dataset: **millions** of images from diverse categories

Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Data-Efficient GANs

GANs Heavily Deteriorate Given Limited Data

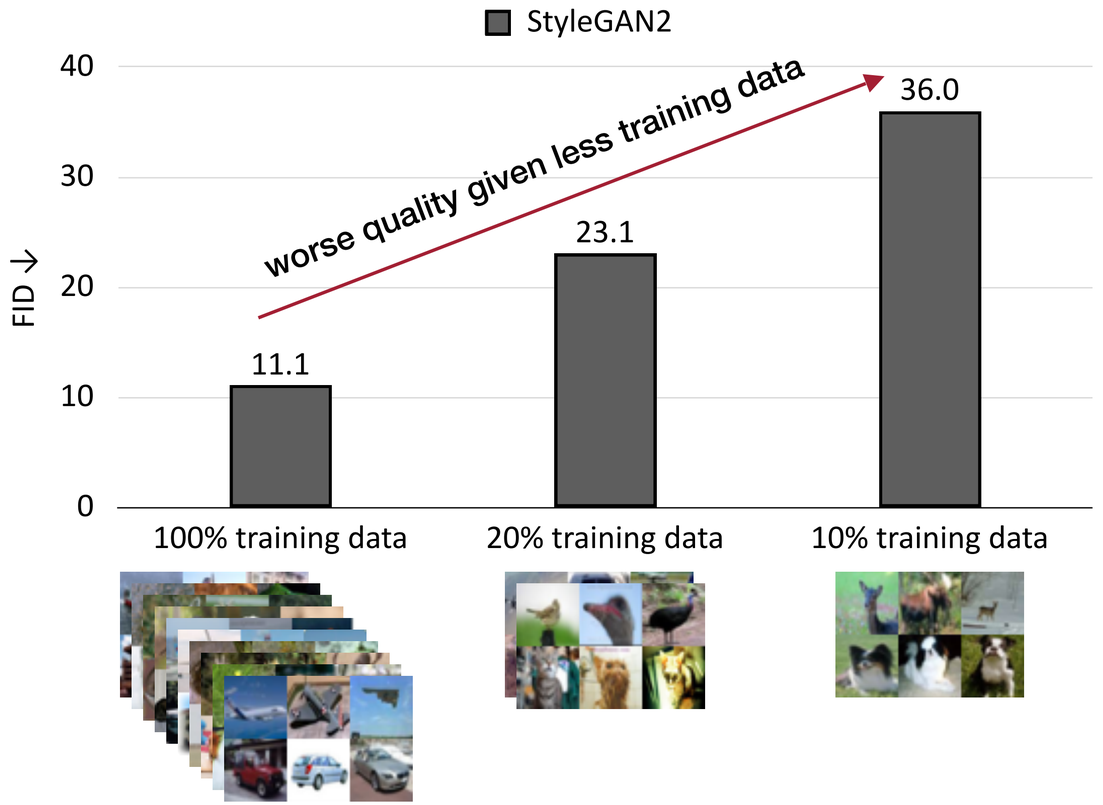
- The current techniques of GAN training lead to inferior results when data is limited
- Generated samples of StyleGAN2. The quality is poor given limited data.

Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Data-Efficient GANs

GANs Heavily Deteriorate Given Limited Data

- The current techniques of GAN training lead to inferior results when data is limited

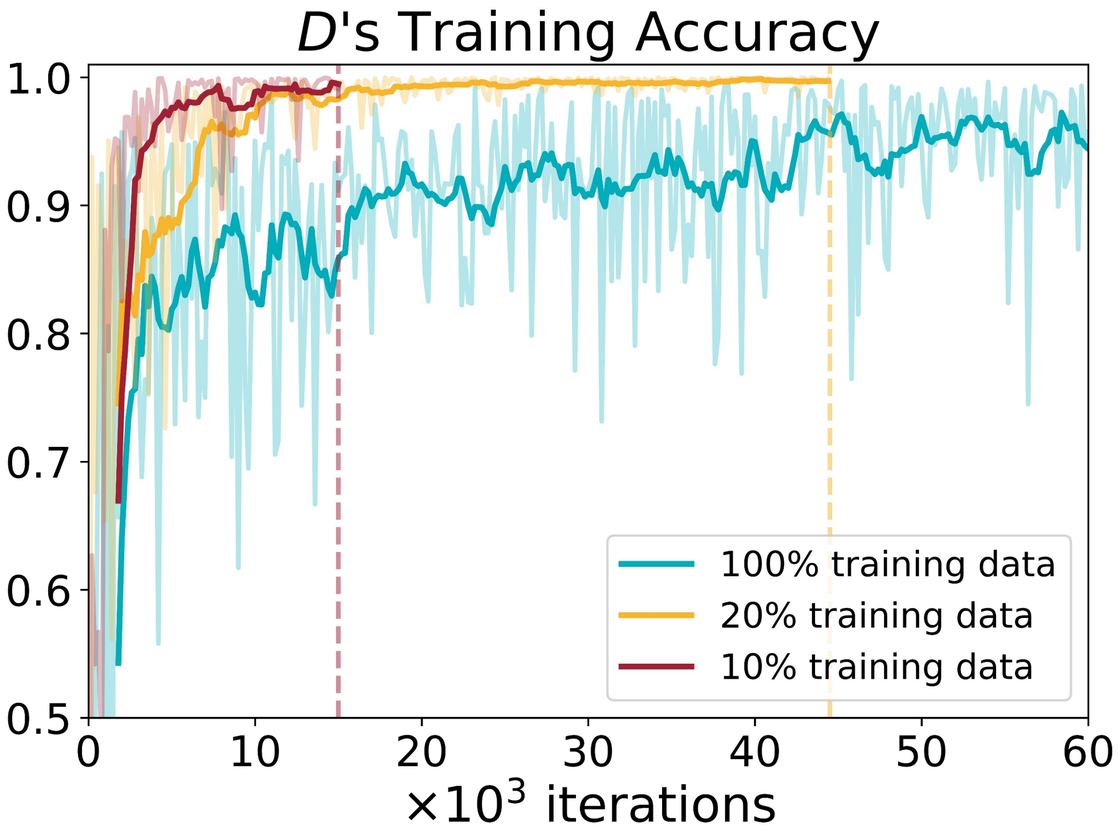


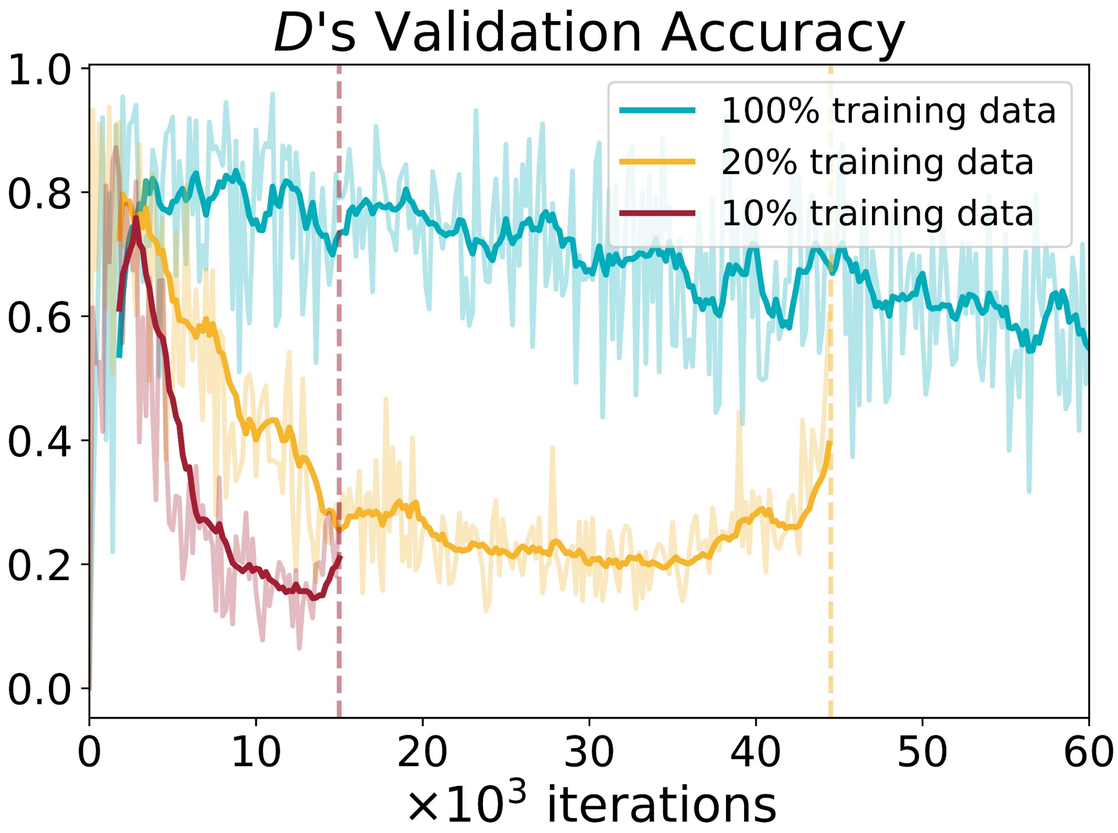
Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Data-Efficient GANs

GANs Heavily Deteriorate Given Limited Data

- We find that the discriminator overfits with less data, leading to worse generator performance



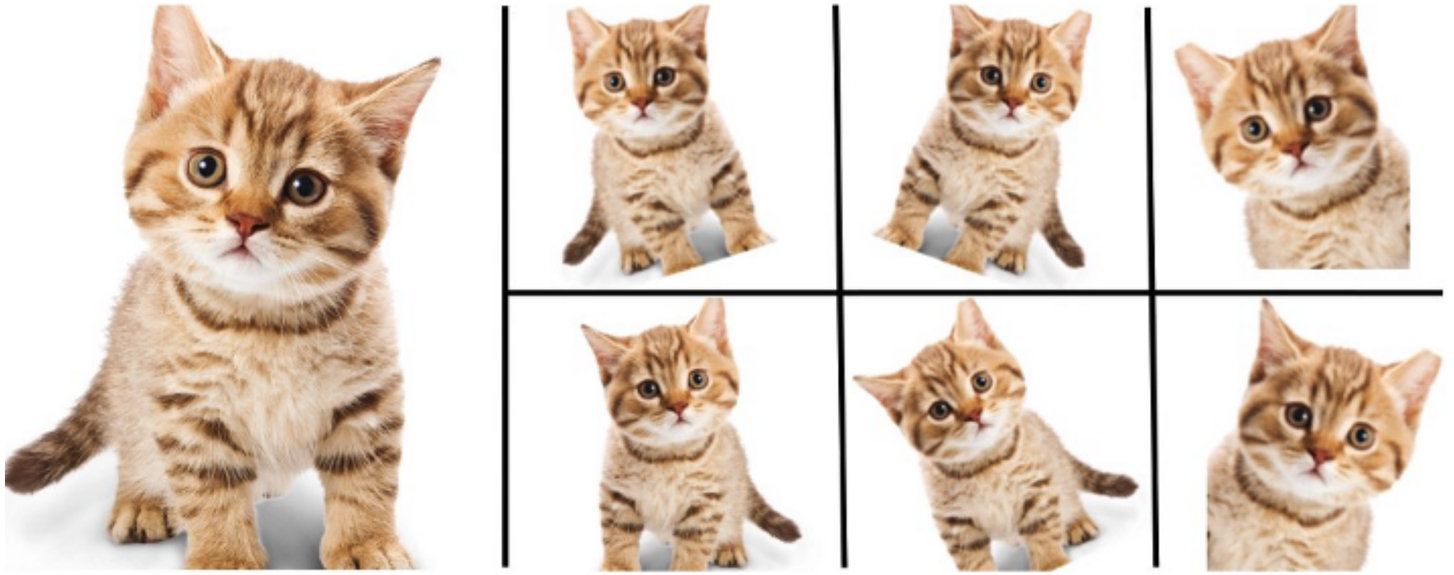


Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Data-Efficient GANs

GANs Heavily Deteriorate Given Limited Data

- We find that the discriminator overfits with less data, leading to worse generator performance
- Traditionally, **data augmentation** is effective for overfitting
- **Question**: How to augment GANs?

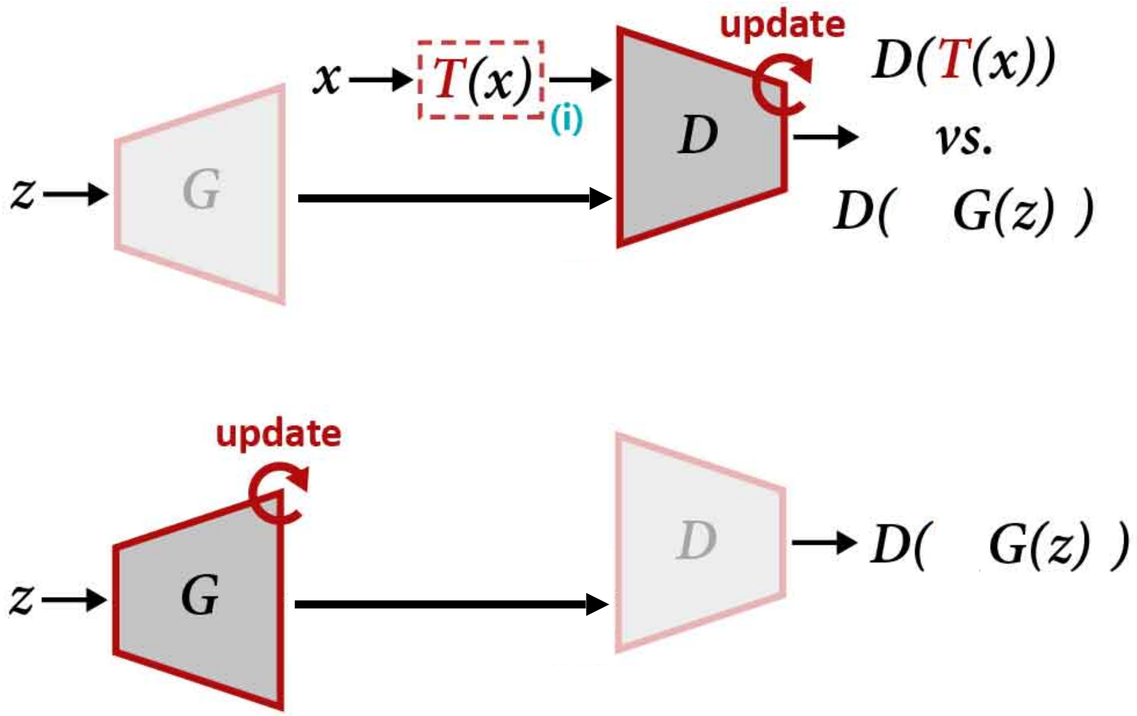


Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

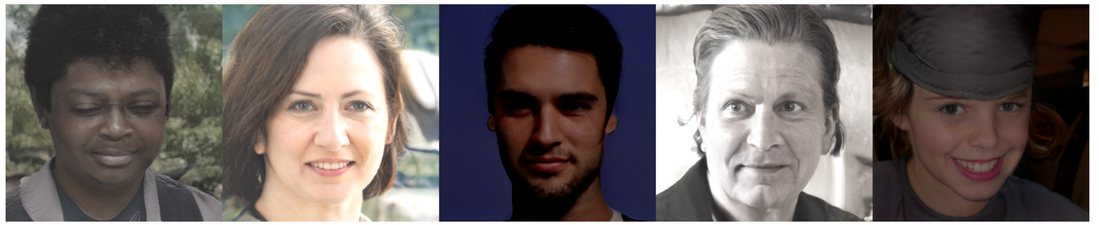
Differentiable Augmentation

Approach 1: Augment reals only

- **Problem**: the same artifacts appear on the generated images.



Generated images



Artifacts from Color jittering

Artifacts from Translation

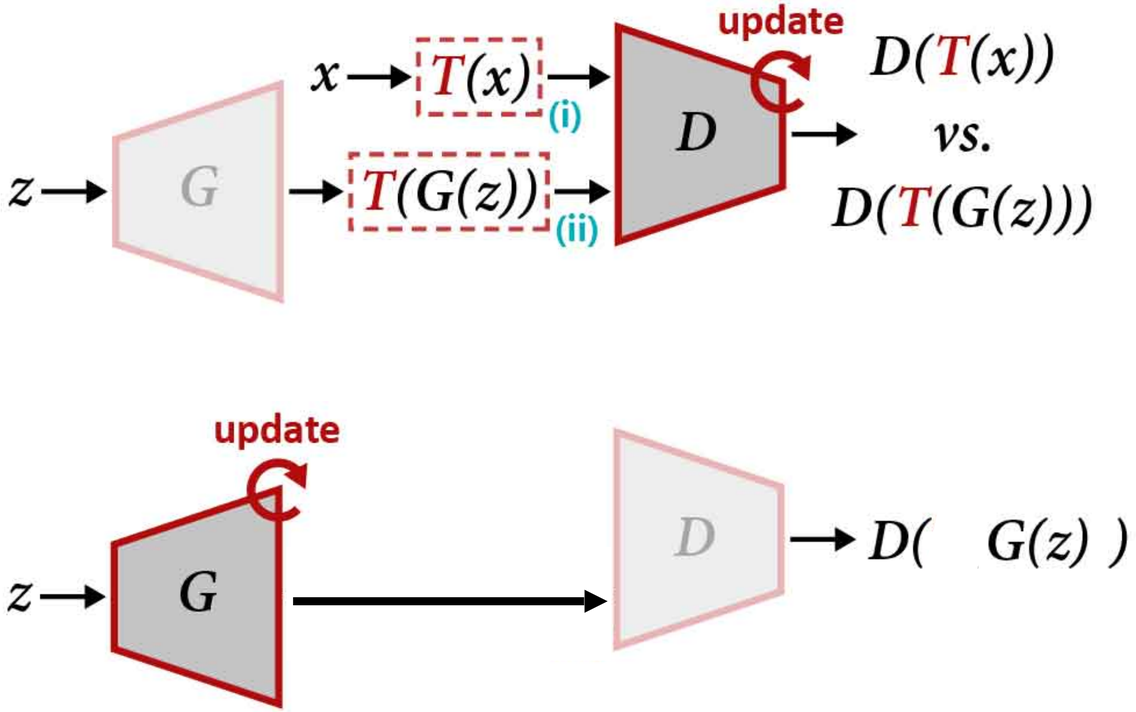
Artifacts from Cutout (DeVries et al.)

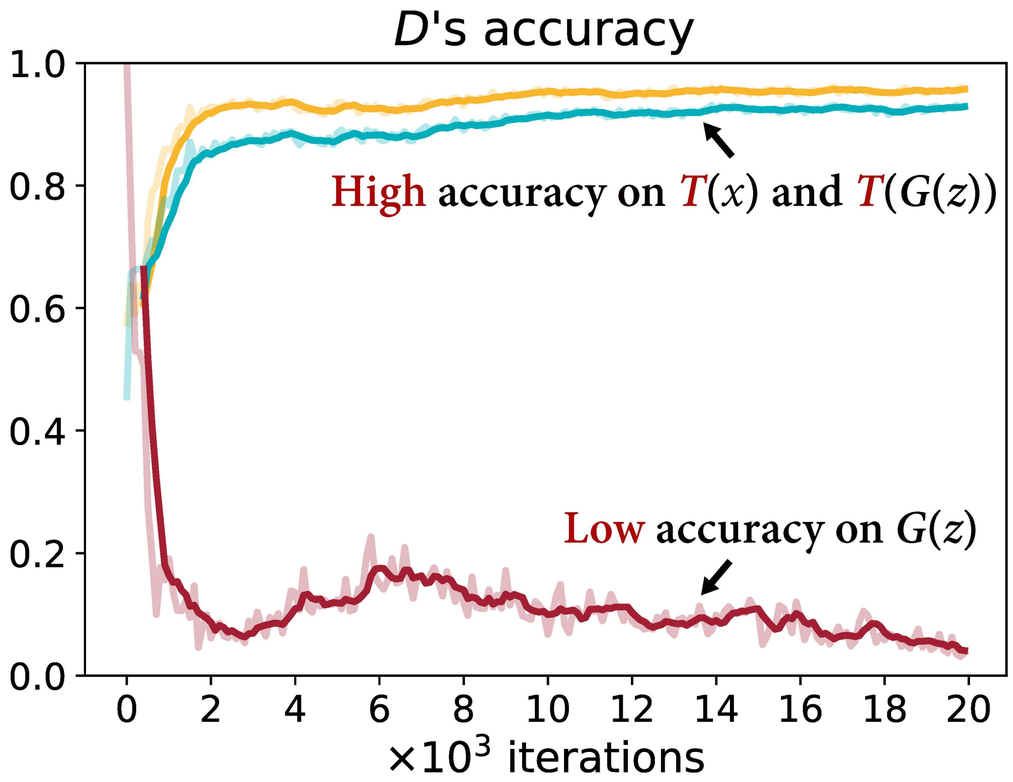
Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Differentiable Augmentation

Approach 2: Augment reals & fakes for discriminator only

- **Problem**: the unbalanced optimization cripples training.



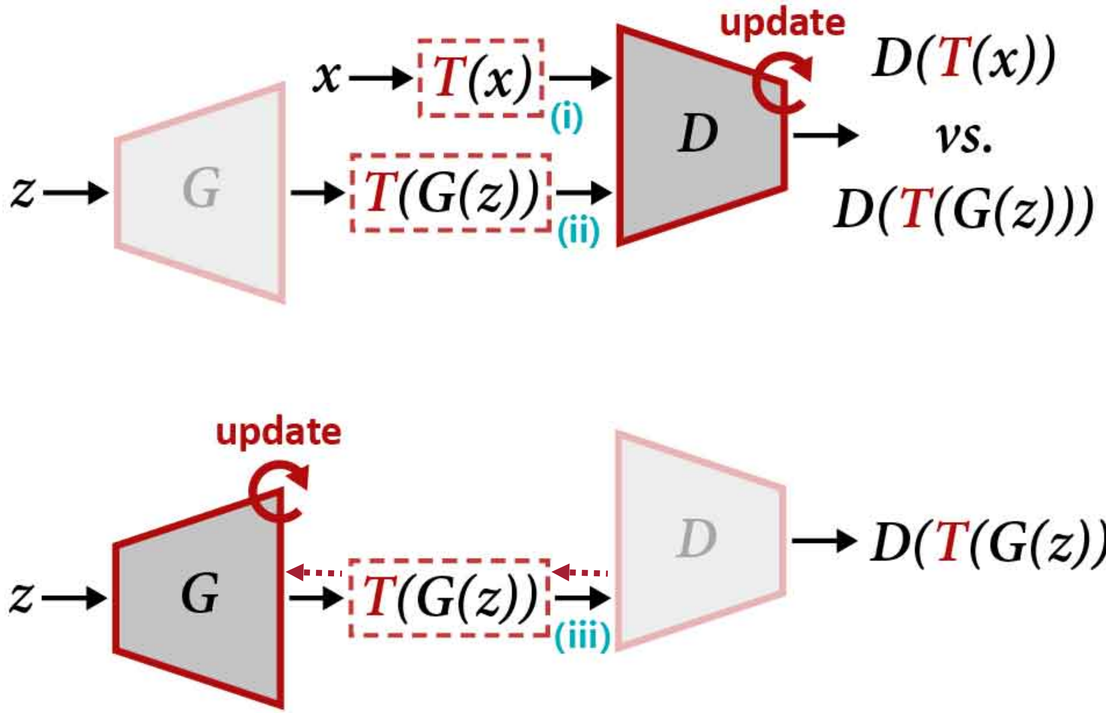


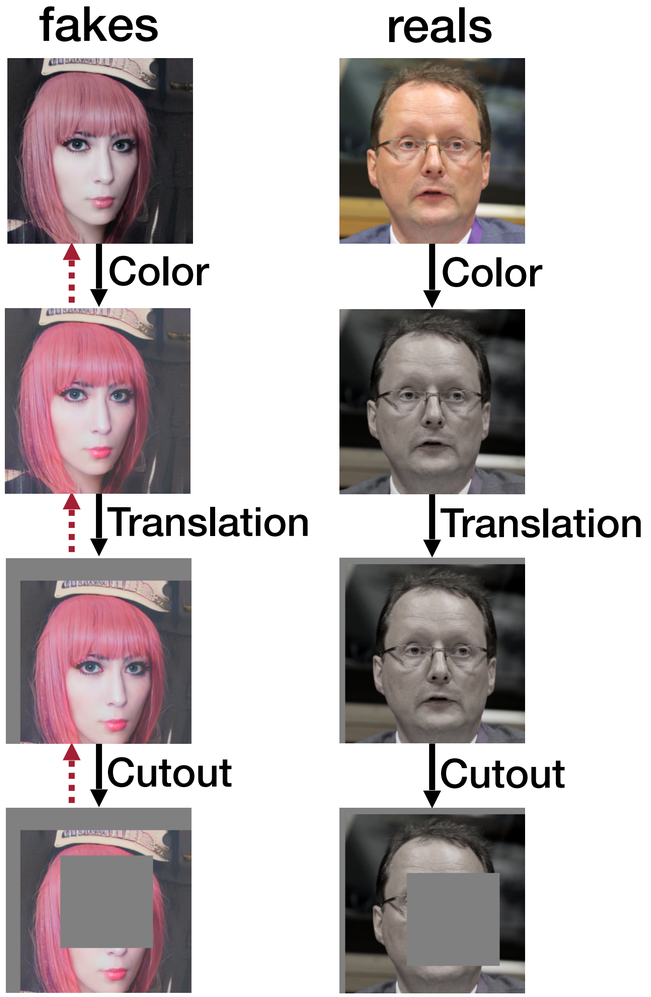
Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Differentiable Augmentation

Approach 3: Differentiable Augmentation (ours)

- Augment reals + fakes for both D and G



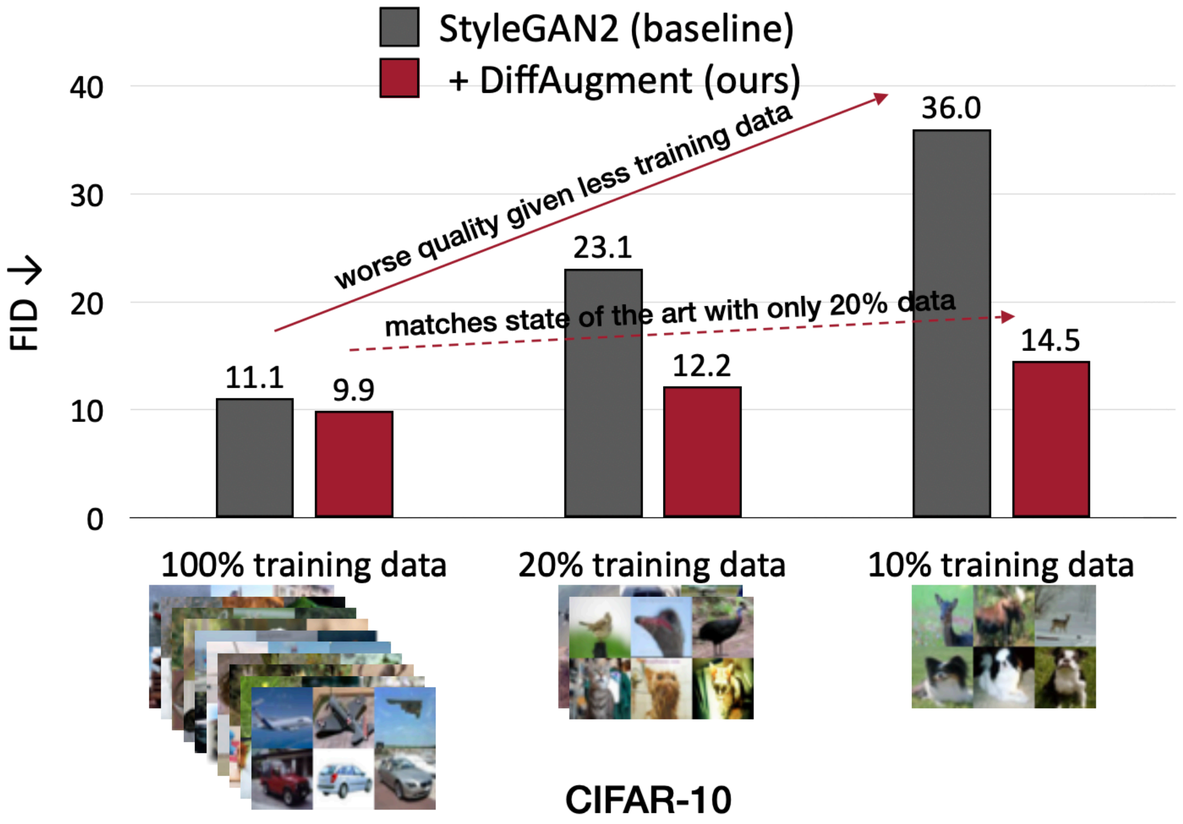


Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Differentiable Augmentation

Results

- Improving GAN training, especially with fewer data



Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Differentiable Augmentation

Results

- Low-shot generation

Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]

Differentiable Augmentation

Results

- 100-shot interpretation.
- The smooth interpolation results suggest little overfitting of our method, even given only **100** images

Summary of Today's Lecture

In this lecture, we introduce:

1. Efficient video understanding
    1. 2D CNNs for video understanding
    2. 3D CNNs for video understanding
    3. Temporal Shift Module (TSM)
    4. Other efficient methods for video understanding
2. Efficient generative models
    1. GAN Compression (compress generators with NAS+distillation)
    2. Anycost GAN (dynamic cost vs. quality trade-off)
    3. Spatially Sparse Inference (save computation by only updating edited regions)
    4. Differentiable Augmentation (data efficient training of GANs)

In the next lecture, we will introduce:

1. Efficient Transformers

References

1. A Review on Deep Learning Techniques for Video Prediction [Oprea et al., 2020]
2. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition [Wang et al. 2016]
3. Two-Stream Convolutional Networks for Action Recognition in Videos [Simonyan et al., 2014]
4. Long-term Recurrent Convolutional Networks for Visual Recognition and Description [Donahue et al., 2016]
5. Learning Spatiotemporal Features with 3D Convolutional Networks [Tran et al. 2014]
6. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset [Carreira et al. 2018]
7. TSM: Temporal Shift Module for Efficient Video Understanding [Lin et al., 2019]
8. Efficient Convolutional Network for Online Video Understanding [Zolfaghari et al., 2018]
9. Non-local Neural Networks [Wang et al., 2017]
10. Training kinetics in 15 minutes: Large-scale distributed training on videos [Lin et al., 2019]
11. A Closer Look at Spatiotemporal Convolutions for Action Recognition [Tran et al., 2018]
12. SlowFast Networks for Video Recognition [Feichtenhofer et al., 2019]
13. X3D: Expanding Architectures for Efficient Video Recognition [Feichtenhofer et al., 2020]

References

14. SCSampler: Sampling Salient Clips from Video for Efficient Action Recognition [Korbar et al., 2019]
15. Recurrent Residual Module for Fast Inference in Videos [Pan et al., 2018]
16. Generative Adversarial Networks [Goodfellow et al., 2014]
17. Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]
18. Hierarchical Text-Conditional Image Generation with CLIP Latents [Ramesh et al., 2022]
19. GAN Compression: Learning Efficient Architectures for Conditional GANs [Li et al., 2020]
20. Image-to-Image Translation with Conditional Adversarial Networks [Isola et al., 2016]
21. Semantic Image Synthesis with Spatially-Adaptive Normalization [Park et al., 2019]
22. Anycost gans for interactive image synthesis and editing [Lin et al., 2021]
23. Efficient Spatially Sparse Inference for Conditional GANs and Diffusion Models [Li et al., 2022]
24. SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations [Meng et al., 2022]
25. Differentiable Augmentation for Data-Efficient GAN Training [Zhao et al., 2020]