Diffusion Models Beat GANs on Image Synthesis

Final Presentation for Seminar
on Selected Topics in Machine

Speaker: Shukrullo Nazirjonov

M.Sc in Autonomy Technologies

shukrullo.nazirjonov@fau.d

17-Jan-2025

Overview

Generative models

Brief overview on DDPMs

Improved DDPMs

Key paper contributions

Post-mortem analysis

Figures from Midjourney Gallery

Generative Models

An incomplete list:

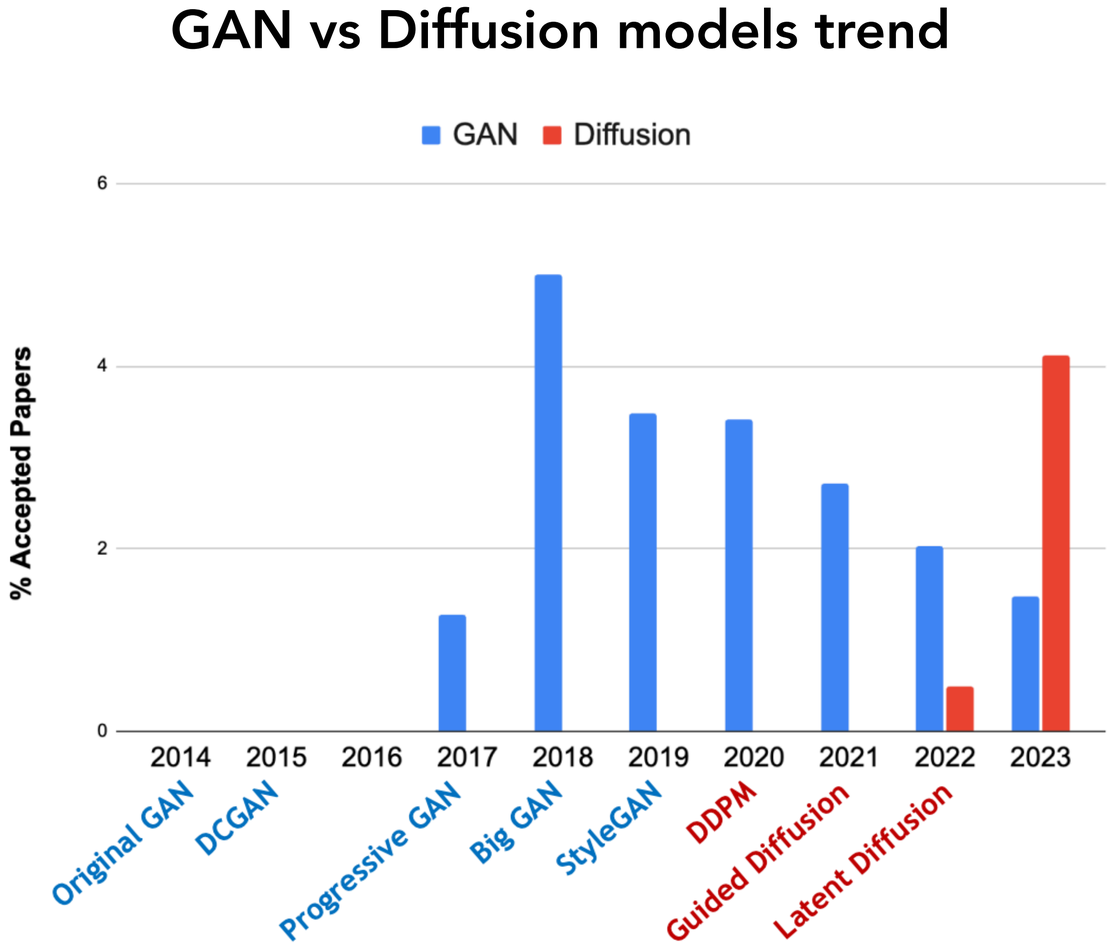
Autoregressive models (PixelRNN, GPT)

Latent variable models (VAE, VQ-VAE)

Flow-based models (Normalizing Flows)

GANs

Diffusion Models (kind of mixture of above)



Figures from [CVPR 2023 tutorial on Diffusion Models]

Sohl-Dickstein et al., 2015

Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Jascha Sohl-Dickstein JASCHA@STANFORD.EDU
Stanford University

Eric A. Weiss EAWEISS@BERKELEY.EDU
University of California, Berkeley

Niru Maheswaranathan NIRUM@STANFORD.EDU
Stanford University

Surya Ganguli SGANGULI@STANFORD.EDU
Stanford University

Song et al., 2019

Generative Modeling by Estimating Gradients of the Data Distribution

Yang Song
Stanford University
yangsong@cs.stanford.edu

Stefano Ermon
Stanford University
ermon@cs.stanford.edu

Ho et al., 2020

Denoising Diffusion Probabilistic Models

Jonathan Ho
UC Berkeley
jonathanho@berkeley.edu

Ajay Jain
UC Berkeley
ajayj@berkeley.edu

Pieter Abbeel
UC Berkeley
pabbeel@cs.berkeley.edu

Denoising Diffusion Probabilistic Models (DDPMs)

Forward process: add Gaussian noise to data

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} x_{t-1}, \sqrt{\beta_t} I)$$

β_t is a variance scheduler

Reverse process: "remove" noise —> learn by DNN:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1};\ \mu_\theta(x_t, t),\ \Sigma_\theta(x_t, t))$$

$\Sigma_\theta(x_t, t)$ can be fixed or learned

$p_\theta(x_{t-1} \mid x_t) \approx q(x_{t-1} \mid x_t)$ (true reverse process approximation)

Denoising Diffusion Probabilistic Models (DDPMs)

DDPM Training

$\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon)$ is just x_t computed with reparametrization trick

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: **until** converged

How much noise was **added** to obtain x_t from x_0?

DDPM Sampling

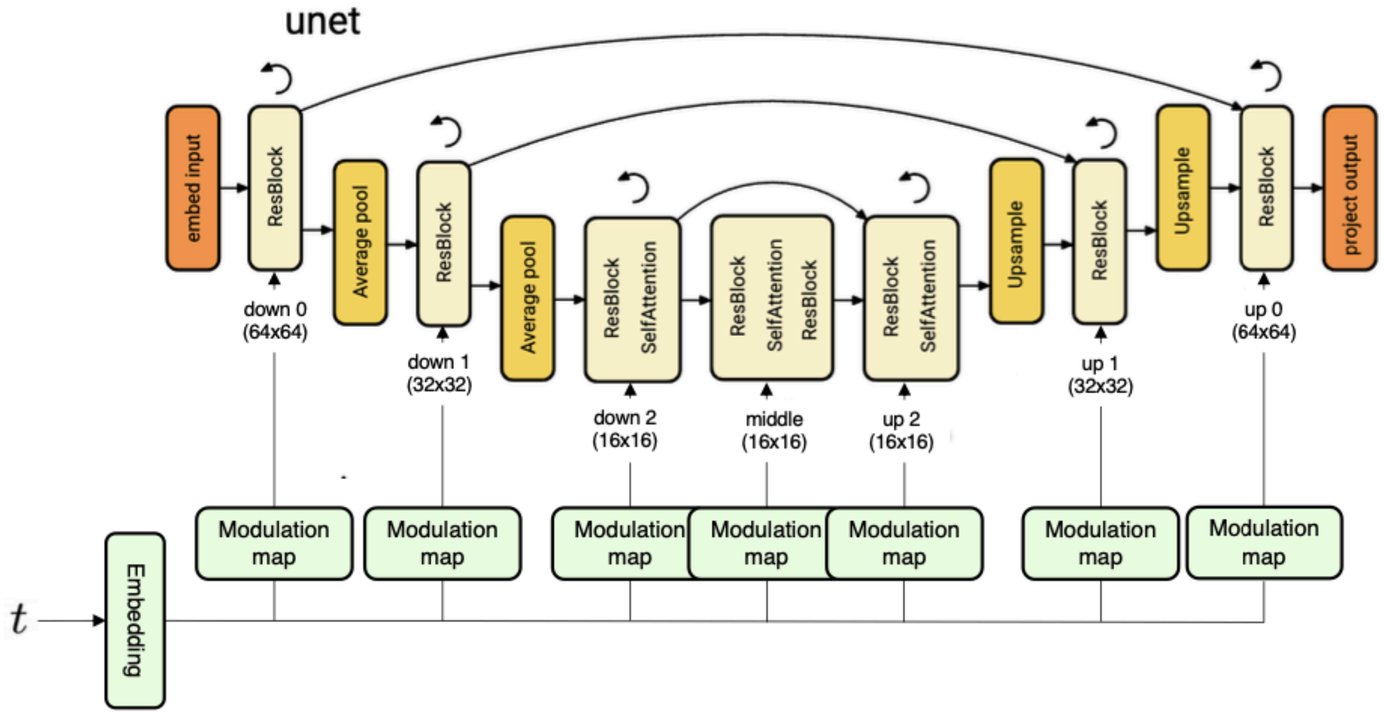
Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Iteratively remove predicted noise for T steps (follow the Markov chain in reverse)

Figures modified from Ho et al., 2020

DDPM network architecture



U-Net model with ResNet blocks and self-attention layers.

t is added on each ResNet block via sinusoidal positional encoding, following Vaswani et al., 2017

Figure from Hoogeboom et al., 2023

Common Metrics on Generative Modeling

Metric	Definition	Ideal Value
Log Likelihood	Data explanation capability	Higher
Precision	Image fidelity/ resemblance to training data	Higher
Recall	Diversity / distribution coverage	Higher
FID	Image quality	Lower
Inception score	Diversity&class confidence	Higher

No single metric tells the full story

Check [“Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models”]() **for details.**

Improving log-likelihood

Improved Denoising Diffusion Probabilistic Models

Alex Nichol [*1] Prafulla Dhariwal [*1]

February 2021

Improving other metrics

Diffusion Models Beat GANs on Image Synthesis

Prafulla Dhariwal*
OpenAI
prafulla@openai.com

Alex Nichol*
OpenAI
alex@openai.com

May 2021

Learned Reverse Process Variance $\Sigma_\theta(x_t, t)$

Reverse Diffusion process: $p_\theta(x_{t-1}|x) = \mathcal{N}(x_{t-1};\ \mu_\theta(x_t, t),\ \Sigma_\theta(x_t, t))$

DDPM: $\Sigma_\theta(x_t, t)$ was fixed, either set to β_t or $\bar{\beta}_t$.

Nichol and Dhariwal [2021] proposed to learn $\Sigma_\theta(x_t, t)$ as an interpolation between β_t and $\bar{\beta}_t$ in log-space:

$$\Sigma_\theta(x_t, t) = \exp(v \log \beta_t + (1 - v)\log \bar{\beta}_t)$$

Better log-likelihood !

Loss function

$$L_{simple} = E_{t,x_0,\epsilon}[||\epsilon - \epsilon_\theta(x_t, t)||^2]$$

$$L_{hybrid} = L_{simple} + \lambda L_{vlb}$$

Model	ImageNet	CIFAR
Glow (Kingma & Dhariwal, 2018)	3.81	3.35
Flow++ (Ho et al., 2019)	3.69	3.08
PixelCNN (van den Oord et al., 2016c)	3.57	3.14
SPN (Menick & Kalchbrenner, 2018)	3.52	-
NVAE (Vahdat & Kautz, 2020)	-	2.91
Very Deep VAE (Child, 2020)	3.52	2.87
PixelSNAIL (Chen et al., 2018)	3.52	2.85
Image Transformer (Parmar et al., 2018)	3.48	2.90
Sparse Transformer (Child et al., 2019)	3.44	**2.80**
Routing Transformer (Roy et al., 2020)	**3.43**	-
DDPM (Ho et al., 2020)	3.77	3.70
DDPM (cont flow) (Song et al., 2020b)	-	2.99
Improved DDPM (ours)	**3.53**	**2.94**

Figures from Nichol and Dhariwal [2021]

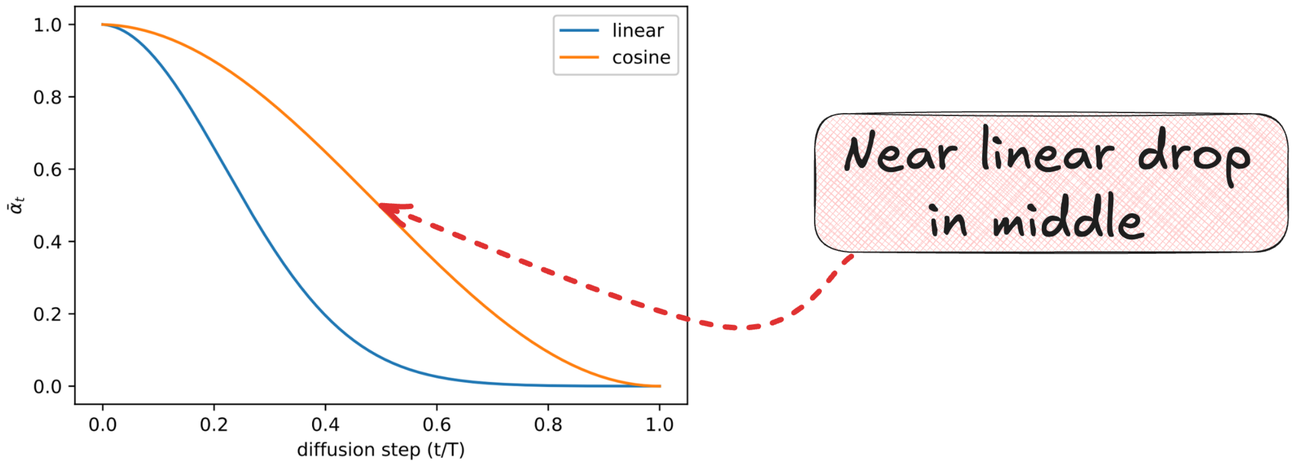
Parametrized noise scheduler β_t

[Nichol and Dhariwal [2021]] proposed to use a different variance schedule to set β_t:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \text{ where } f(t) = \cos\left(\frac{t/T + s}{1+s}\frac{\pi}{2}\right)^2$$

$\beta_t = \min(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999)$ where s is a small offset to prevent β_t becoming too small.

More progressive forward diffusion process, especially for large values of t



Figures modified from [Nichol and Dhariwal [2021]]

Diffusion Models Beat GANs on Image Synthesis

GANs have an unfair advantage:

Extensive architecture optimization (2014-onwards)

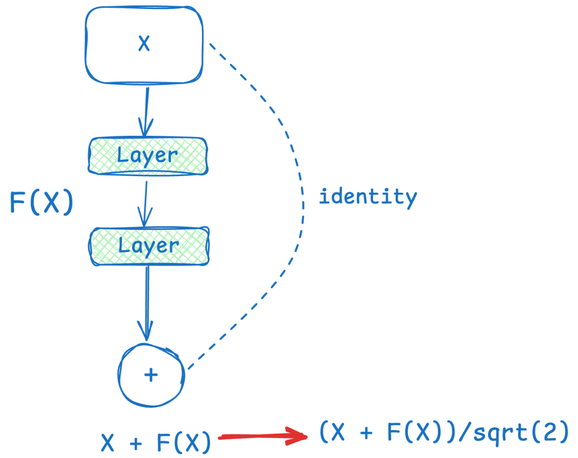
Trade off diversity for **fidelity**

Let's bring those benefits to Diffusion Models!

Ablation

Optimizes U-Net architecture to learn $\epsilon_\theta(x_t, t)$ by ablating on:

- depth vs width
- number of attention heads and sizes
- BigGAN residual blocks
- Adaptive Group Normalization
- rescaling residual connections with $1/\sqrt{2}$



Channels	Depth	Heads	Attention resolutions	BigGAN up/downsample	Rescale resblock	FID 700K	FID 1200K
160	2	1	16	✗	✗	15.33	13.21
128	4					-0.21	-0.48
		4				-0.54	-0.82
			32,16,8			-0.72	-0.66
				✓		-1.20	-1.21
					✓	0.16	0.25
160	2	4	32,16,8	✓	✗	**-3.14**	**-3.00**

Number of heads	Channels per head	FID
1		14.08
2		-0.50
4		-0.97
8		-1.17
	32	-1.36
	64	-1.03
	128	-1.08

Architectural ablation results

Figures from Nichol and Dhariwal [2021]

Classifier Guidance

How can we generate a sample x given class information y (🐶) ?

Bayes' rule

$$p(x|y) = \frac{p(x)\ p(y|x)}{p(y)}$$

Log

$$\log p(x|y) = \log p(y|x) + \log p(x) - \log p(y)$$

Gradient

$$\nabla_x \log p(x|y) = \nabla_x \log p(y|x) + \nabla_x \log p(x)$$

Scaled guidance

$$\nabla_x \log_\gamma(x|y) = \gamma \nabla_x \log p(y|x) + \nabla_x \log p(x)$$

conditional score function

classifier gradient

unconditional score function

The classifier $p(y|x) = p_\phi(y|x_t)$ is trained on noisy images.

Classifier Guidance: Intuition

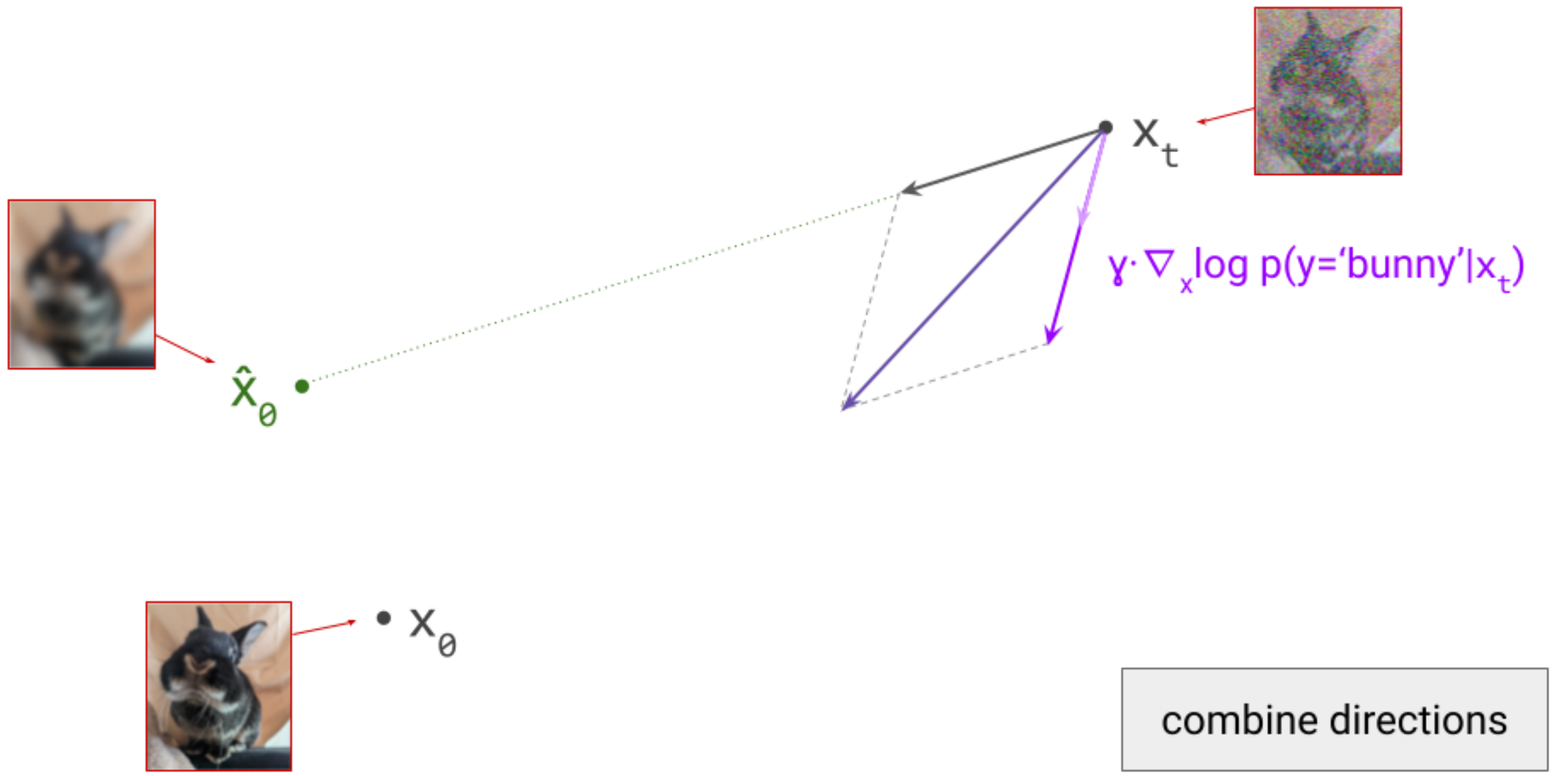


Figure from Sander Dieleman

Classifier Guidance

y = "Pembroke Welsh corgi"

s=1

More diverse/ lower quality

Less diverse/ higher quality

s=10

Benefits:

Better FID score (image quality/fidelity)

Controlled image generation: replace class label with another modality (image, text, etc).

Figures from Nichol and Dhariwal [2021]

More samples

BigGAN

Diffusion

Training Set

Figures from Nichol and Dhariwal [2021]

What we know as of 2024

Jonathan Ho & Tim Salimans, 2022

Problem: difficult to obtain noise-robust classifier

Solution: classifier-free guidance

Classifier guidance: $\nabla_x \log p(x \mid y) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y \mid x)$

Classifier-Free guidance: $\nabla_x \log p(x \mid y) = (1-\gamma) \nabla_x \log p(x) + \gamma \nabla_x \log p(x \mid y)$

More and more advanced guidance methods: ControlNet, GLIDE, DALL-E 2 (classifier guidance+CLIP), Imagen (classifier guidance + T5)

What we know as of 2024

- Cosine scheduler was better but ***even better*** noise schedulers exist

On the Importance of Noise Scheduling for Diffusion Models

Ting Chen
Google Research, Brain Team
iamtingchen@google.com

- Faster sampling methods via distillation

Consistency Models

Yang Song [1] **Prafulla Dhariwal** [1] **Mark Chen** [1] **Ilya Sutskever** [1]

ONE STEP DIFFUSION VIA SHORTCUT MODELS

Kevin Frans
UC Berkeley
kvfrans@berkeley.edu

Danijar Hafner
UC Berkeley

Sergey Levine
UC Berkeley

Pieter Abbeel
UC Berkeley

1. Guidance: a cheat code for diffusion models: https://sander.ai/2022/05/26/guidance.html

2. The geometry of diffusion guidance : https://sander.ai/2023/08/28/geometry.html

3. DDPM - Diffusion Models Beat GANs on Image Synthesis: YouTube video

4. Deep Unsupervised Learning CS294-158-SP24 UC Berkeley course notes

Thank you for your $softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)$!