StoryGAN:
A Sequenital Conditional GAN for Story Visualization

Yitong Li, Zhe Gan, Yelong Shen, Jingjing Liu, Yu Cheng, Yuexin Wu, Lawrence Carin, David Carlson and Jianfeng Gao

Reviewer: Byeong Jo Kim

Introduction

Story Visualization task

to **generate a sequence of images** to **describe a story** written in a **multi-sentence paragraph**

Two Challenge

the sequence of images must **consistently** and **coherently** depict the whole story

compared with image generation

coherent image sequence

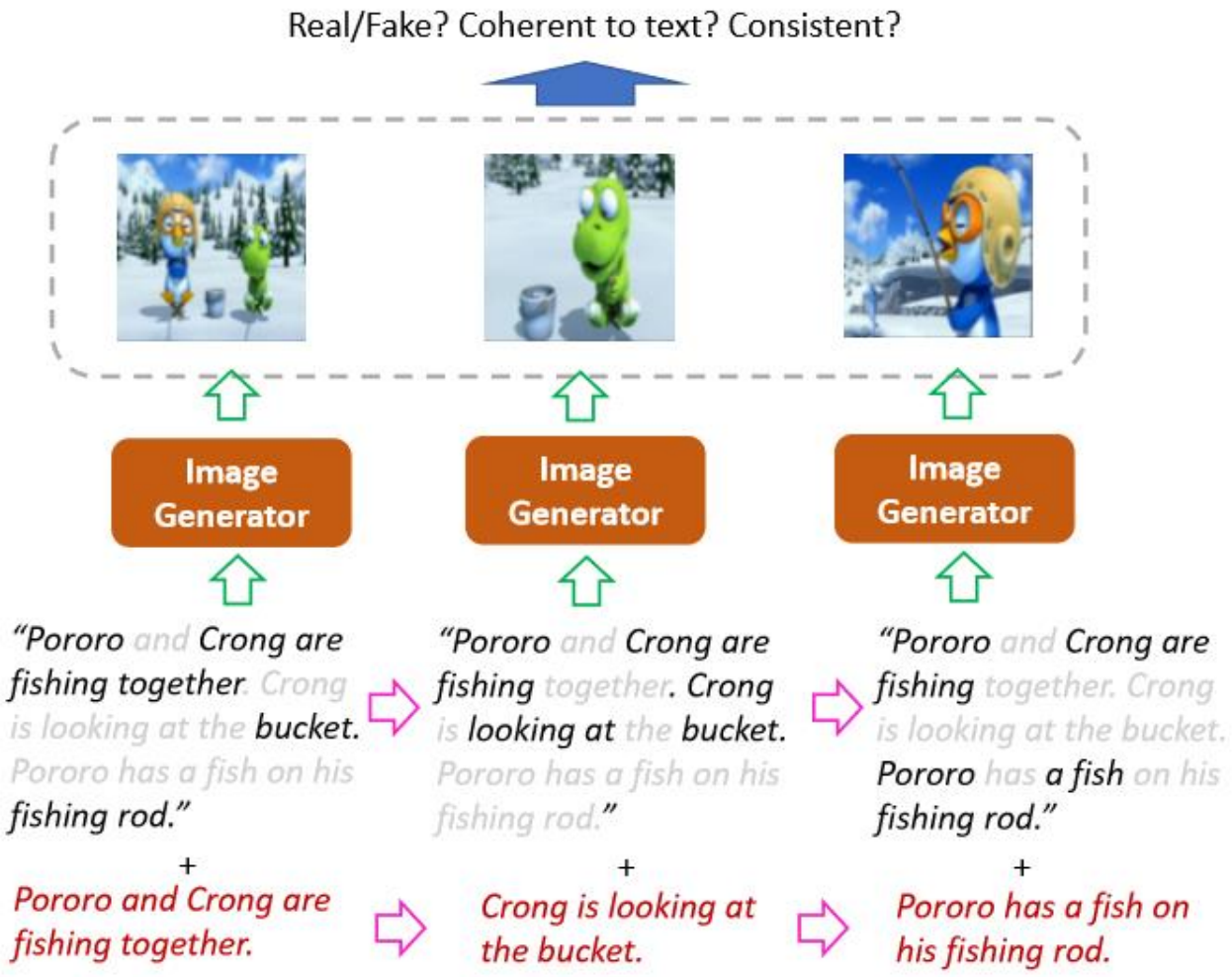
display the logic of the storyline

compared with video generation

smooth motion transitions is not important

sharp change of scenes

StoryGAN


Real/Fake? Coherent to text? Consistent?
Image Generator
Image Generator
Image Generator
"Pororo and Crong are fishing together. Crong is looking at the bucket. Pororo has a fish on his fishing rod."
"Pororo and Crong are fishing together. Crong is looking at the bucket. Pororo has a fish on his fishing rod."
"Pororo and Crong are fishing together. Crong is looking at the bucket. Pororo has a fish on his fishing rod."
+
+
+
Pororo and Crong are fishing together.
Crong is looking at the bucket.
Pororo has a fish on his fishing rod.

StoryGAN

D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. **Universal sentence encoder**. arXiv preprint arXiv:1803.11175, 2018. 3, 7

128-d

$$S = [\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_T]$$

one generated image per sentence
with **locally** and **globally consistent**

sentence – image story – images

$$\hat{\boldsymbol{X}} = [\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \cdots, \hat{\boldsymbol{x}}_T]$$

StoryGAN

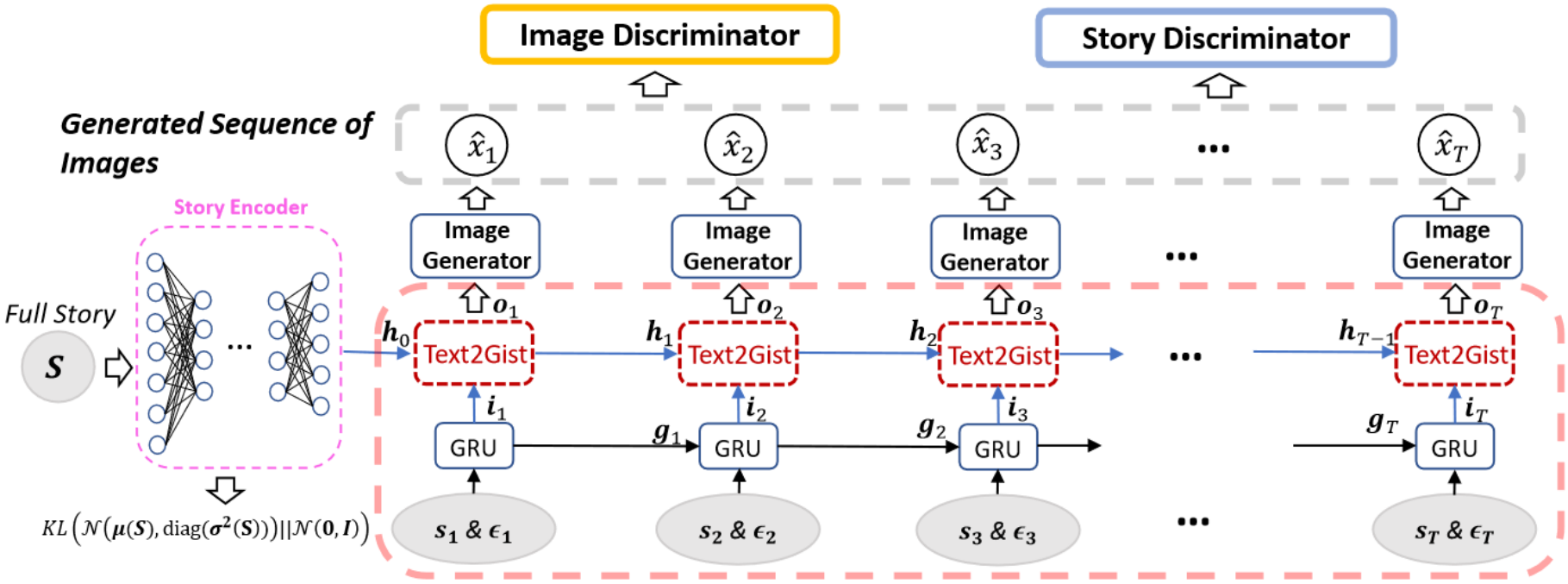
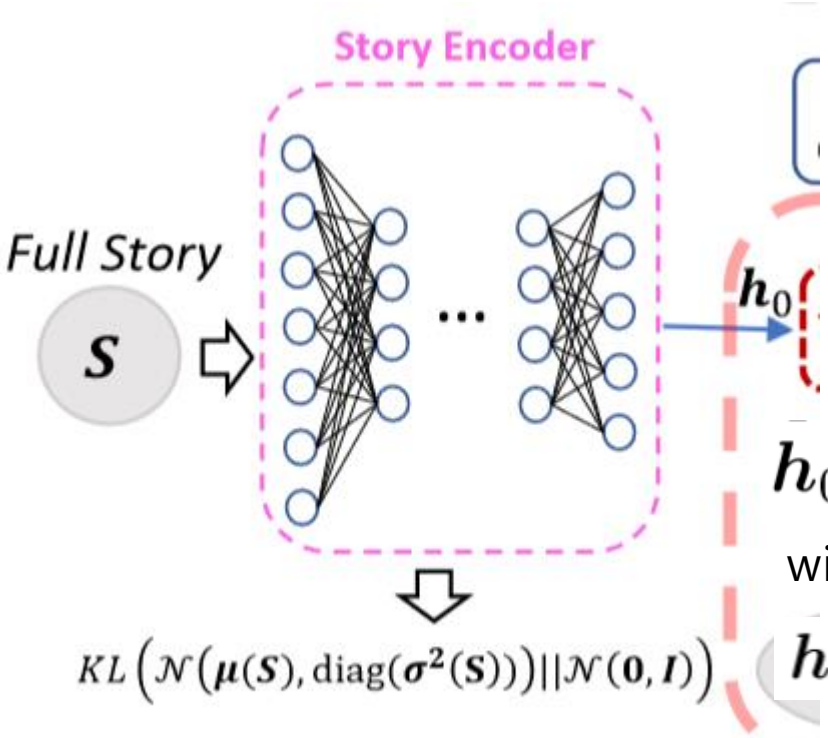


Image Discriminator
Story Discriminator
Generated Sequence of Images
Story Encoder
Full Story
S
Image Generator
Text2Gist
GRU
$\hat{x}_1$
$\hat{x}_2$
$\hat{x}_3$
$\hat{x}_T$
o_1
o_2
o_3
o_T
h_0
h_1
h_2
h_{T-1}
i_1
i_2
i_3
i_T
g_1
g_2
g_T
$s_1 \& \epsilon_1$
$s_2 \& \epsilon_2$
$s_3 \& \epsilon_3$
$s_T \& \epsilon_T$
$KL\left(\mathcal{N}(\boldsymbol{\mu}(\boldsymbol{S}), \mathrm{diag}(\boldsymbol{\sigma}^2(\boldsymbol{S}))) || \mathcal{N}(\mathbf{0}, \boldsymbol{I})\right)$

StoryGAN – Story Encoder



$$S = [\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_T]$$

E( )

$$\boldsymbol{h}_0$$

$$\boldsymbol{h}_0 \sim E(\boldsymbol{S}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{S}), \boldsymbol{\Sigma}(\boldsymbol{S}))$$

with reparameterization trick

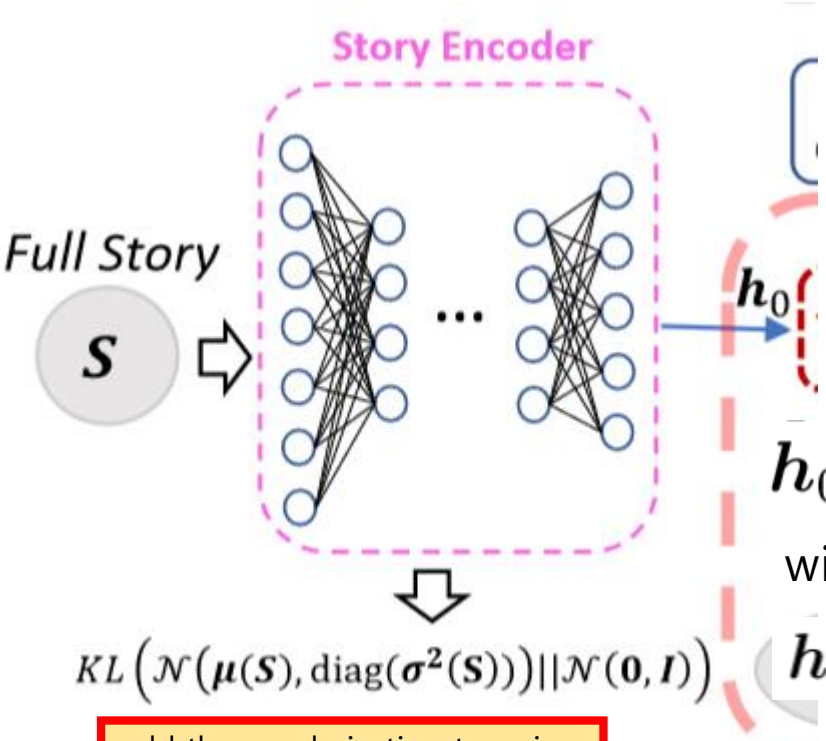
$$\boldsymbol{h}_0 = \boldsymbol{\mu}(\boldsymbol{S}) + \boldsymbol{\sigma}^2(\boldsymbol{S})^{\frac{1}{2}} \odot \boldsymbol{\epsilon}_S$$

$$\boldsymbol{\epsilon}_S \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

leading to a compact, semantic **representation** of S
adding **randomness** to the generation process

$$\mathcal{L}_{KL} = KL\left(\mathcal{N}\left(\boldsymbol{\mu}(\boldsymbol{S}), \mathrm{diag}(\boldsymbol{\sigma}^2(\boldsymbol{S}))\right) || \mathcal{N}\left(\mathbf{0}, \boldsymbol{I}\right)\right)$$

StoryGAN – Story Encoder



$$S = [\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_T]$$

E( )

$$\boldsymbol{h}_0$$

$$\boldsymbol{h}_0 \sim E(\boldsymbol{S}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{S}), \boldsymbol{\Sigma}(\boldsymbol{S}))$$

with reparameterization trick

$$\boldsymbol{h}_0 = \boldsymbol{\mu}(\boldsymbol{S}) + \boldsymbol{\sigma}^2(\boldsymbol{S})^{\frac{1}{2}} \odot \boldsymbol{\epsilon}_S$$

$$\boldsymbol{\epsilon}_S \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

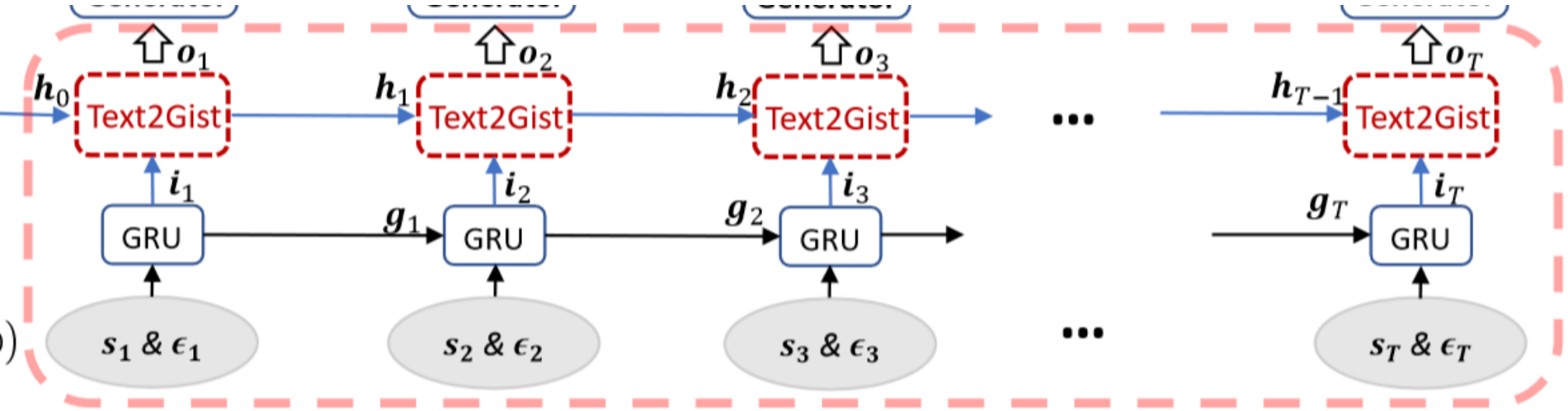
le … semantic **representation** of S
ac … o the generation process

add the regularization term in

H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In ICCV, 2017. 1, 2, 3, 4, 6

$$\mathcal{L}_{KL} = KL\left(\mathcal{N}\left(\boldsymbol{\mu}(\boldsymbol{S}), \text{diag}(\boldsymbol{\sigma}^2(\boldsymbol{S}))\right) || \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})\right)$$

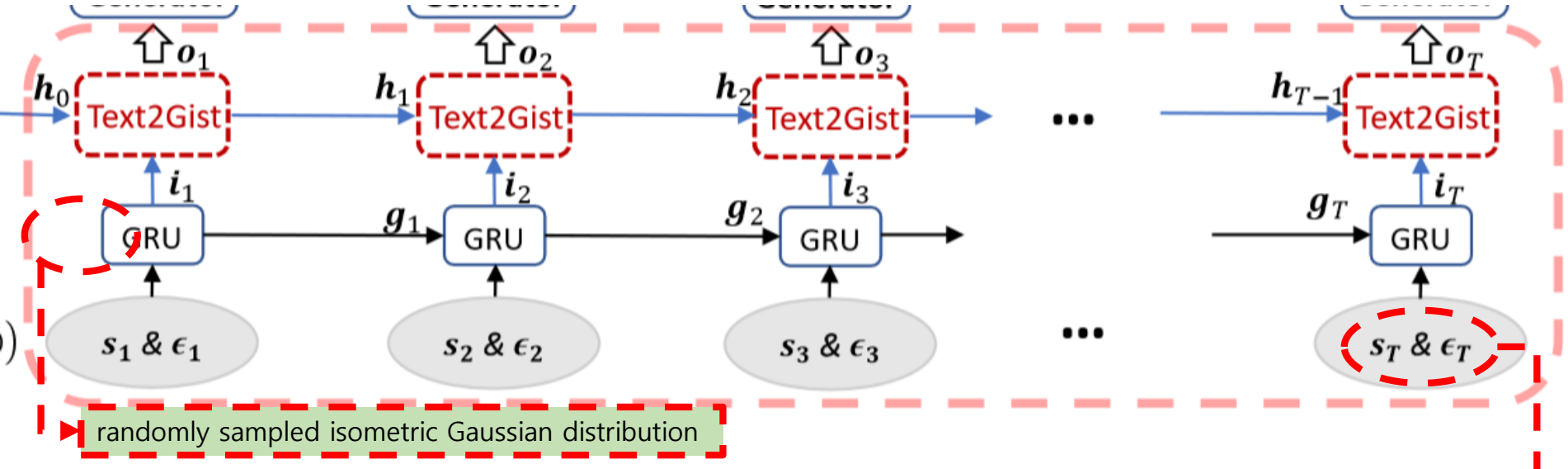
StoryGAN – Context Encoder



- how to update the **contextual information** to effectively capture background changes
- how to combine **new inputs** and **random noise** when generating each image, to visualize the change of characters, which may shift dramatically

-> **RNN based Context Encoder**

StoryGAN – Context Encoder



randomly sampled isometric Gaussian distribution

- GRU + Text2Gist

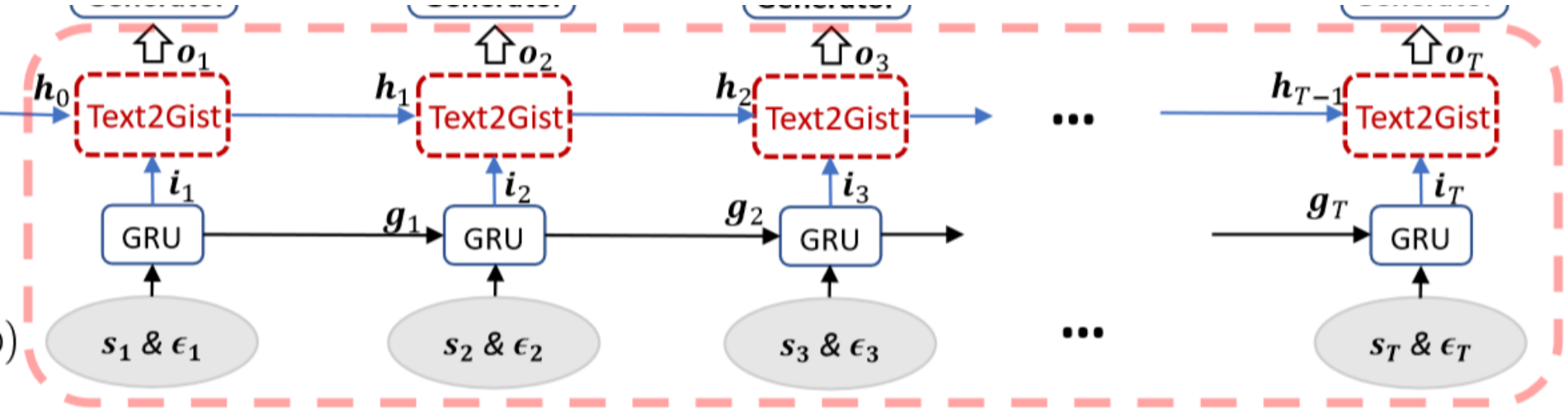
concatenation of sentence and isometric Gaussian noise

$$\boldsymbol{i}_t, \boldsymbol{g}_t = \mathrm{GRU}(\boldsymbol{s}_t, \boldsymbol{\epsilon}_t, \boldsymbol{g}_{t-1}),$$

$$\boldsymbol{o}_t, \boldsymbol{h}_t = \mathrm{Text2Gist}(\boldsymbol{i}_t, \boldsymbol{h}_{t-1}).$$

"Gist" vector: combines all the global and local context information

StoryGAN – Context Encoder



update gate

decide how much information from the previous step should be kept

reset gate

determine what to forget from h

$$\boldsymbol{z}_t = \sigma_z\left(\boldsymbol{W}_z\boldsymbol{i}_t + \boldsymbol{U}_t\boldsymbol{h}_{t-1} + \boldsymbol{b}_z\right),$$

$$\boldsymbol{r}_t = \sigma_r\left(\boldsymbol{W}_r\boldsymbol{i}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r\right),$$

$$\boldsymbol{h}_t = (\mathbf{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \sigma_h\left(\boldsymbol{W}_h\boldsymbol{i}_t + \boldsymbol{U}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h\right),$$

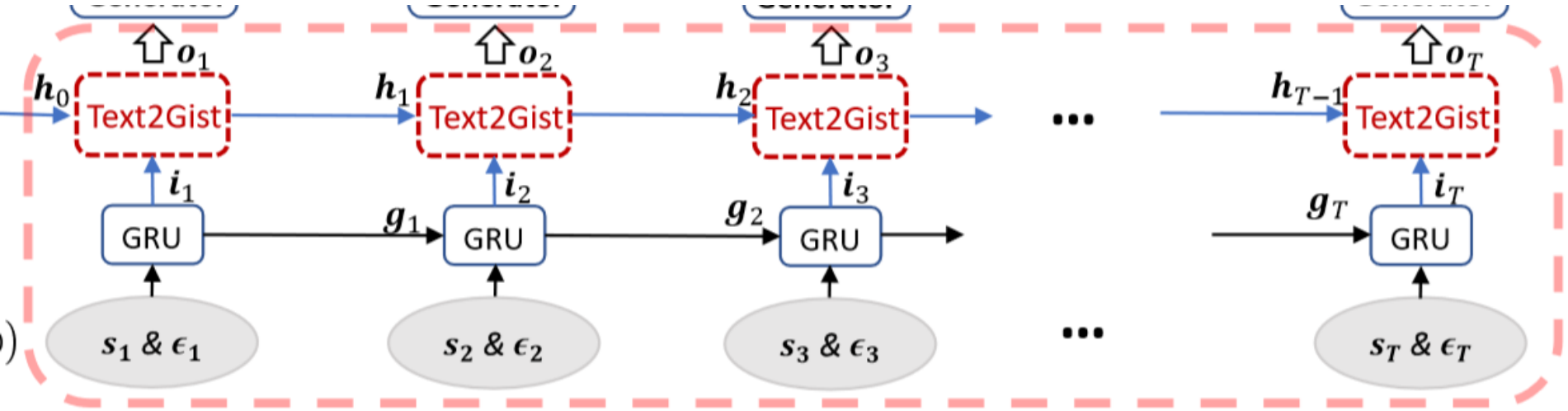
$$\boldsymbol{o}_t = \mathrm{Filter}(\boldsymbol{i}_t) * \boldsymbol{h}_t,$$

Convolution

Filter(): transforms vector i to a multi-channel filter 1D filter

$C_{out} \times 1 \times 1 \times \mathrm{len}(\boldsymbol{h}_t)$

StoryGAN – Context Encoder



update gate

decide how much information from the previous step should be kept

reset gate

determine what to forget from h

Convolution

Filter(): transforms vector i_t to a multi-channel filter 1D filter $C_{out} \times 1 \times 1 \times \text{len}(\boldsymbol{h}_t)$

i_t: from sentence
h_t: from story

infuse the global contextual information from h_t and local formation from i_t
help s_t to pick out the important part from the story

$$\boldsymbol{o}_t = \text{Filter}(\boldsymbol{i}_t) * \boldsymbol{h}_t,$$

StoryGAN – Discriminators

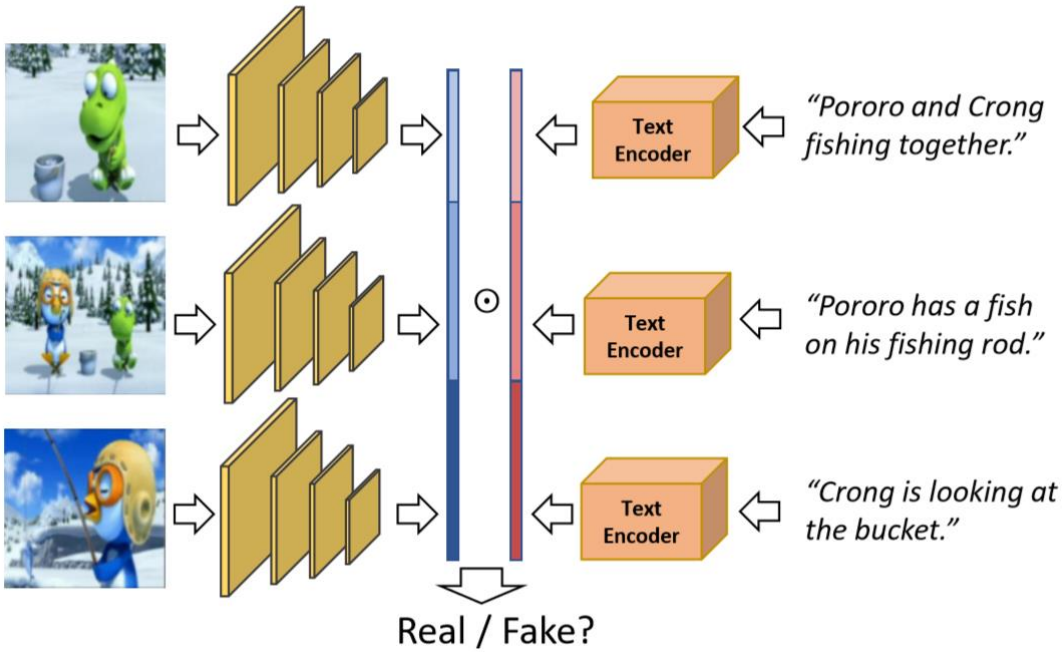
Image Discriminator

$$\{s_t, h_0, \hat{x}_t\} \Longleftrightarrow \{s_t, h_0, x_t\}$$

- the **same sentence** can have a significantly **different generated image** depending on the **context**,
- so it is important to give the **encoded context information** to the **discriminator**

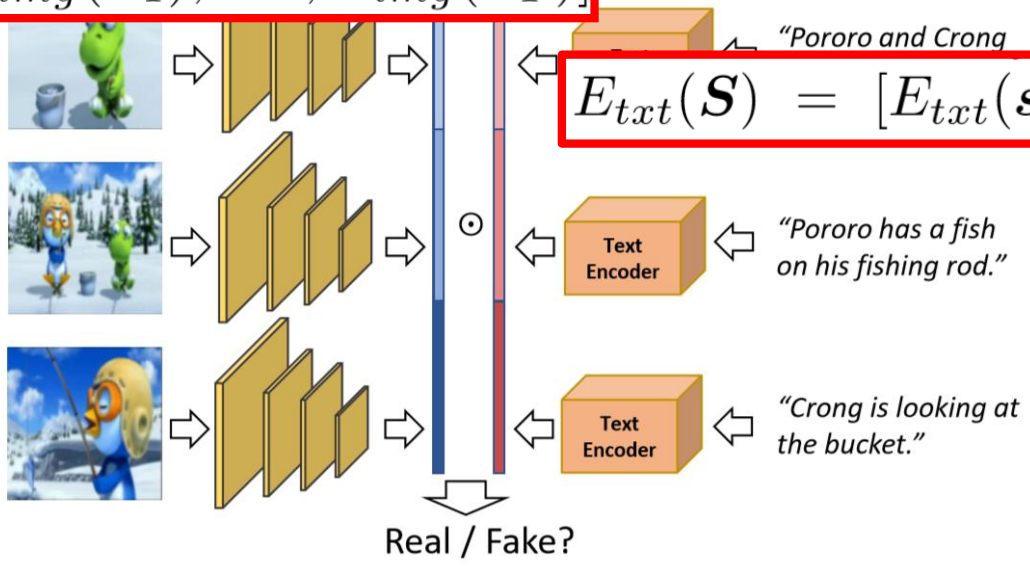
Story Discriminator

- enforce the global consistency of the generated image sequence given story S



StoryGAN – Discriminators

$$E_{img}(\boldsymbol{X}) = [E_{img}(\boldsymbol{x}_1), \cdots, E_{img}(\boldsymbol{x}_T)]$$



$$D_S = \sigma\left(\boldsymbol{w}^\intercal \left(E_{img}(\boldsymbol{X}) \odot E_{txt}(\boldsymbol{S})\right) + b\right)$$

By **pairing each sentence and image**, the story discriminator can consider both **local matching** and **global consistency jointly**

StoryGAN – Objective function

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\psi}_I, \boldsymbol{\psi}_S} \alpha \mathcal{L}_{Image} + \beta \mathcal{L}_{Story} + \mathcal{L}_{KL}$$

$$\mathcal{L}_{Image} = \sum_{t=1}^{T} (\mathbb{E}_{(\boldsymbol{x}_t, \boldsymbol{s}_t)} [\log D_I(\boldsymbol{x}_t, \boldsymbol{s}_t, \boldsymbol{h}_0; \boldsymbol{\psi}_I)] + \mathbb{E}_{(\boldsymbol{\epsilon}_t, \boldsymbol{s}_t)} [\log(1 - D_I(G(\boldsymbol{\epsilon}_t, \boldsymbol{s}_t; \boldsymbol{\theta}), \boldsymbol{s}_t, \boldsymbol{h}_0; \boldsymbol{\psi}_I))])$$

$$\mathcal{L}_{Story} = \mathbb{E}_{(\boldsymbol{X}, \boldsymbol{S})} [\log D_S(\boldsymbol{X}, \boldsymbol{S}; \boldsymbol{\psi}_S)] + \mathbb{E}_{(\boldsymbol{\epsilon}, \boldsymbol{S})} \left[\log(1 - D_S([G(\boldsymbol{\epsilon}_t, \boldsymbol{s}_t; \boldsymbol{\theta})]_{t=1}^{T}), \boldsymbol{S}; \boldsymbol{\psi}_S))\right]$$

StoryGAN – Algorithm

Algorithm 1 Training Procedure of StoryGAN

Input: Encoded sentence vectors $\boldsymbol{S}_n = [\boldsymbol{s}_{n1}, \boldsymbol{s}_{n2}, \cdots, \boldsymbol{s}_{nT}]$ and corresponding images $\boldsymbol{X}_n = [\boldsymbol{x}_{n1}, \cdots, \boldsymbol{x}_{nT}]$ for $n = 1, \cdots, N$.
Output: Generator parameters $\boldsymbol{\theta}$ and discriminator parameters $\boldsymbol{\psi}_I$ and $\boldsymbol{\psi}_S$.

---

for $iter = 1$ to max_iter **do**
  for $iter_I = 1$ to k_I **do**
    Sample a mini-batch of story-sentence pairs $\{(\boldsymbol{s}_t, \boldsymbol{S}, \boldsymbol{x}_t)\}$ from the training set.
    Compute $\boldsymbol{h}_0$ as the initialization of the Text2Gist layer and the KL regularization term as Eq. (1).
    Generate a single output image $\hat{x}$.
    Update $\boldsymbol{\psi}_I$ and $\boldsymbol{\theta}$.
  end for
  for $iter_S = 1$ to k_S **do**
    Sample a mini-batch of story-image pair $\{(\boldsymbol{S}, \boldsymbol{X})\}$ from training set.
    Compute $\boldsymbol{h}_0$ and update $\boldsymbol{h}_t$ at each time step t
    Generate image sequence $\hat{\boldsymbol{X}}$.
    Update $\boldsymbol{\psi}_S$ and $\boldsymbol{\theta}$.
  end for
end for

image

story

- using **Adam** optimizer
- **different mini-batch sizes** for image and story discriminators

StoryGAN – Experiments

Video generation result is too blurry and not comparable to StoryGAN

- our comparisons are mainly to ablated versions of our proposed model
- for a fair comparison, use same image generator, context encoder and discriminators

ImageGAN: ImageGAN follows the work in [28, 36] and does not use the story discriminator, story encoder and Context Encoder. Each image is generated independently. However, for a reasonable comparison, we concatenate s_t, encoded story S and a noise term as input. Otherwise, the model fails on the task. This is the simplest version of StoryGAN.

SVFN: In "Story Visualization by Filter Network" (SVFN), the concatenation in SVC is replaced by a filter network. Sentence s_t is transformed into a filter and convolved with the encoded story. Specifically, the image generator input is $o_t = \text{Filter}(i_t) * h_0$ instead of Eq. 7.

SVC: In "Story Visualization by Concatenation" (SVC), the Text2Gist cell in StoryGAN is replaced by simple concatenation of the encoded story and description feature vectors [31]. Compared to ImageGAN, SVC includes the additional story discriminator, and is visualized in Figure 4.

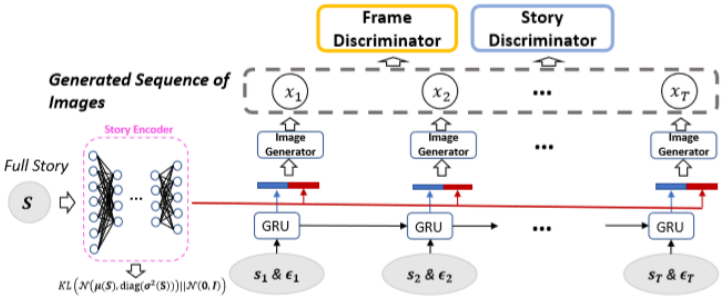


Figure 4: The framework of the baseline model SVC, where the story and individual sentence are concatenated to form the input.

StoryGAN – Experiments with CLEVR-SV

CLEVR -> CLEVR-SV

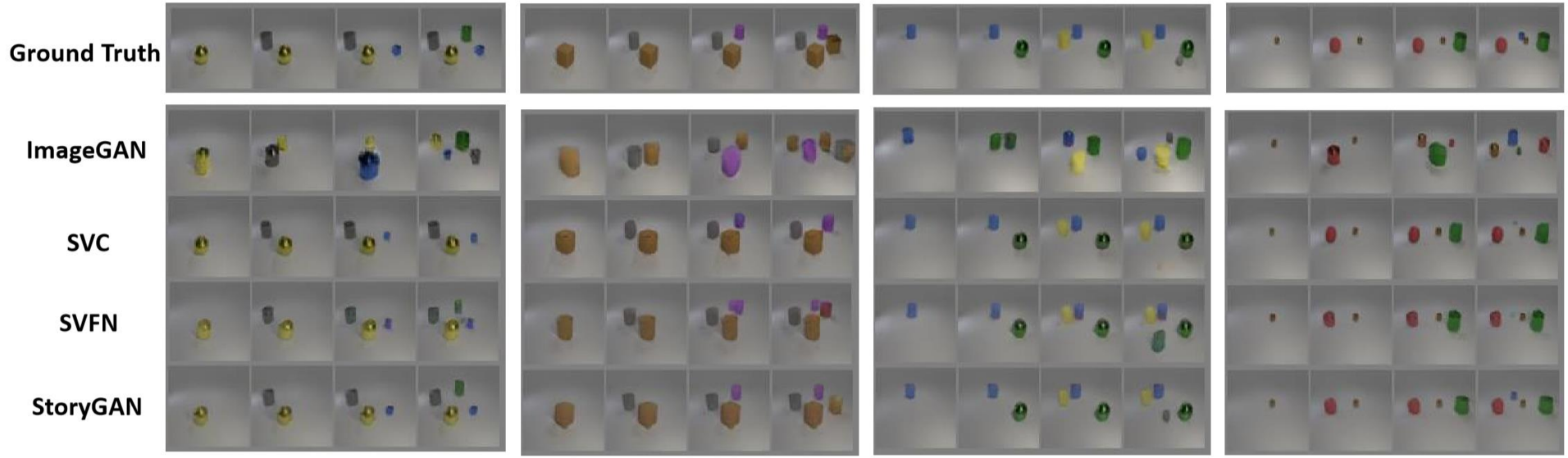
- the **maximum number of objects** in one story is limited to four.
- objects are made of **metallic/rubber** with **eight different colors** and **two different sizes**.
- the **object shape** can be cylinder, cube or sphere.
- the object is **added one at a time**, resulting in a **four-image** sequence per story.

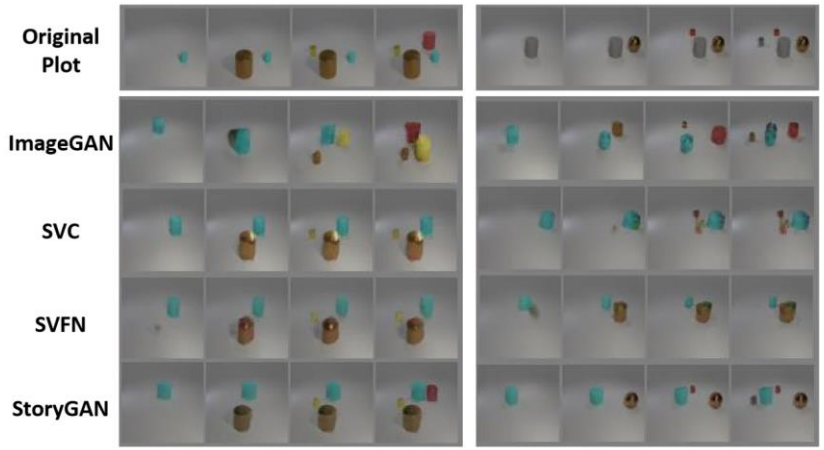
StoryGAN generates more feasible images than others

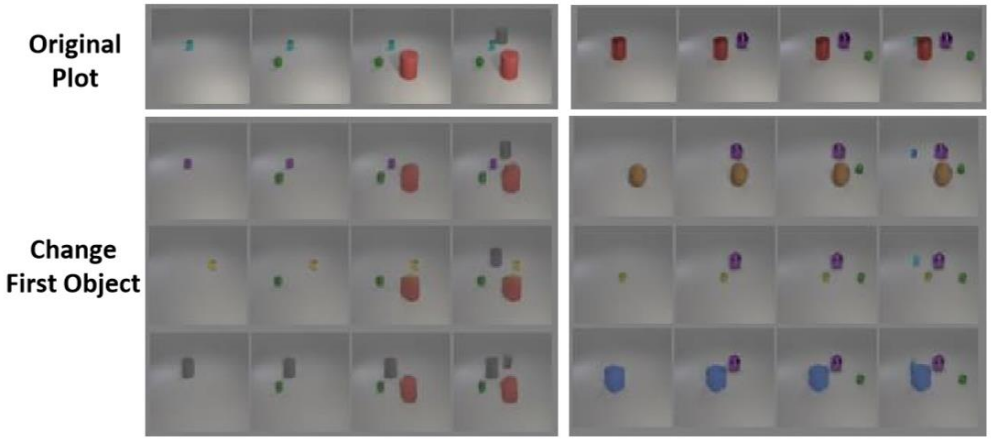
- **Text2Gist** cell tracks the progress of story
- **story and image discriminators** keep the consistency of objects in the generation process
- using the **Story Encoder** to initialize the Text2Gist cell gives better result on first generated image

	ImageGAN [28]	SVC	SVFN	StoryGAN
SSIM	0.596	0.641	0.654	0.672

StoryGAN – Experiments with CLEVR-SV





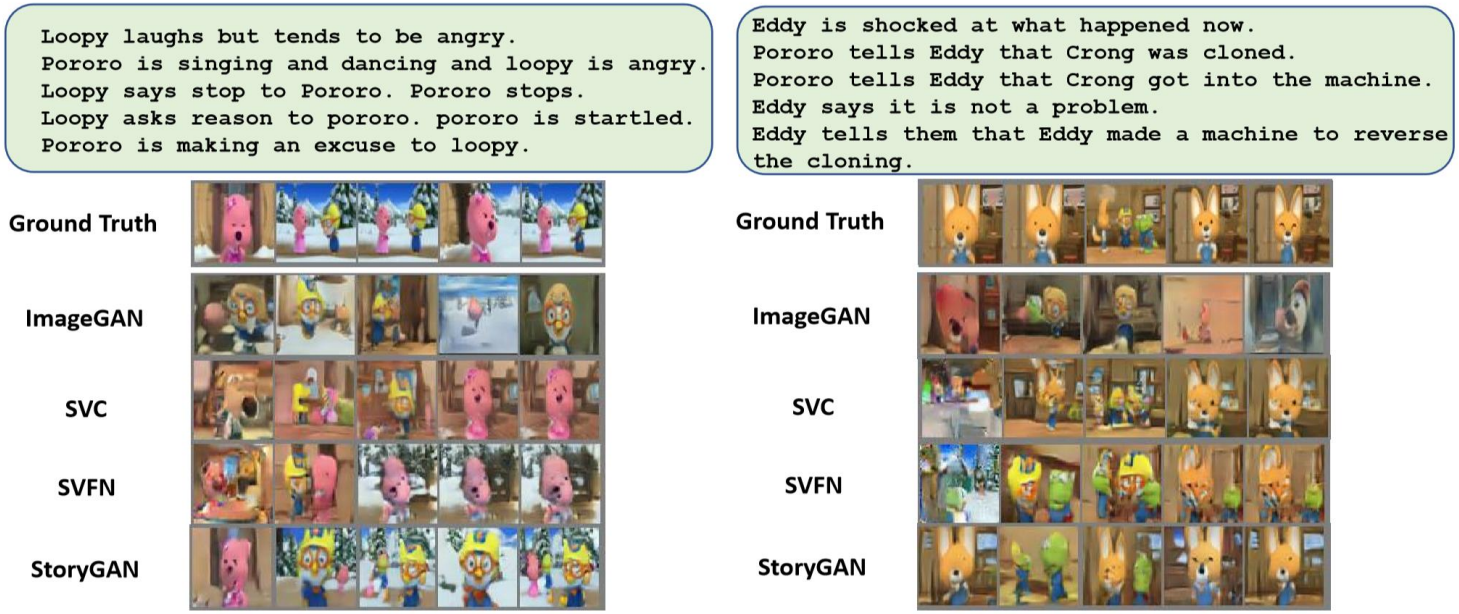


StoryGAN – Experiments with Pororo-SV

Pororo -> Pororo-SV

- randomly pick out one frame during training as the real image sample
- five continuous images form a single story

StoryGAN's **first image** has a much **higher quality** than other baselines -> **Story Encoder**



StoryGAN – Experiments

Input Story: *c1 and c2 are standing in the snow. c1 tells a story to c3. c3 wants to joint c1 and c2. c1 continuous to talk. c1 looks down. They suddenly noticed that there is something lying on the snow.*

C1 = Pororo, C2 = Loppy, C3 = Crong

C1 = Pororo, C2 = Eddy, C3 = Rody

Method	ImageGAN	SVC	SVFN	StoryGAN
Rank	2.91 $\pm$0.05	2.42 $\pm$0.04	2.77 $\pm$0.04	1.94 $\pm$0.05

	StoryGAN vs ImageGAN		
Choice (%)	StoryGAN	ImageGAN	Tie
Visual Quality	74.17 $\pm$1.38	18.60 $\pm$1.38	7.23
Consistence	79.15 $\pm$1.27	15.28 $\pm$1.27	5.57
Relevance	78.08 $\pm$1.34	17.65 $\pm$1.34	4.27

Network Structure

Layer	Story Encoder
1	LINEAR-(128 × T, 128), BN, RELU
Layer	Context Encoder
1	LINEAR-(NOISEDIM + TEXTDIM, 128), BN, RELU
2	GRU-(128, 128)
3	Text2Gist-(128, 128)
Layer	Filter Network
1	LINEAR-(128, 1024), BN, TANH
2	RESHAPE(16, 1, 1, 64)
Layer	Image Generator
1	CONV-(C512, K3, S1, P1), BN, RELU
2	UPSAMPLE-(2,2)
3	CONV-(C256, K3, S1, P1), BN, RELU
4	UPSAMPLE-(2,2)
5	CONV-(C128, K3, S1, P1), BN, RELU
6	UPSAMPLE-(2,2)
7	CONV-(C64, K3, S1, P1), BN, RELU
8	UPSAMPLE-(2,2)
9	CONV-(C3, K3, S1, P1), BN, TANH
Layer	Image Discriminator
1	CONV-(C64, K4, S2, P1), BN, LEAKY RELU
2	CONV-(C128, K4, S2, P1), BN, LEAKY RELU
3	CONV-(C256, K4, S2, P1), BN, LEAKY RELU
4	CONV-(C512, K4, S2, P1),BN, LEAKY RELU
5*	CONV-(C512, K3, S1, P1), BN, LEAKY RELU
6	CONV-(C1, K4, S4, P0), SIGMOID
Layer	Story Discriminator (Image Encoder)
1	CONV-(C64, K4, S2, P1), BN, LEAKY RELU
2	CONV-(C128, K4, S2, P1), BN, LEAKY RELU
3	CONV-(C256, K4, S2, P1), BN, LEAKY RELU
4	CONV-(C512, K4, S2, P1),BN, LEAKY RELU
5	CONV-(C32, K4, S2, P1),BN, CONCAT
6	RESHAPE-(1, 32 × 4 × T)
Layer	Story Discriminator (Text Encoder)
1	LINEAR-(128 × T, 32 × 4 × T), BN