Masked Generative Story Transformer with Character Guidance and Caption Augmentation

Christos Papadimitriou, Giorgos Filandrianos, Maria Lymperaiou, and Giorgos Stamou

Artificial Intelligence and Learning Systems Laboratory
School of Electrical and Computer Engineering
National Technical University of Athens
papadimitri2000@gmail.com, {geofila,marialymp}@islab.ntua.gr,
gstam@cs.ntua.gr

Abstract. Story Visualization (SV) is a challenging generative vision task, that requires both visual quality and consistency between different frames in generated image sequences. Previous approaches either employ some kind of memory mechanism to maintain context throughout an auto-regressive generation of the image sequence, or model the generation of the characters and their background separately, to improve the rendering of characters. On the contrary, we embrace a completely parallel transformer-based approach, exclusively relying on Cross-Attention with past and future captions to achieve consistency. Additionally, we propose a Character Guidance technique to focus on the generation of characters in an implicit manner, by forming a combination of text-conditional and character-conditional logits in the logit space. We also employ a caption-augmentation technique, carried out by a Large Language Model (LLM), to enhance the robustness of our approach. The combination of these methods culminates into state-of-the-art (SOTA) results over various metrics in the most prominent SV benchmark (Pororo-SV), attained with constraint resources while achieving superior computational complexity compared to previous arts. The validity of our quantitative results is supported by a human survey[1] .

Keywords: Story Visualization · Transformers · LLM augmentation

1 Introduction

The task of Story Visualization (SV), introduced in 2019 by Li et al. [11], involves generating a sequence of images, each corresponding to a sentence in a given textual narrative. This task can be regarded as an extension of Text-to-Image Generation, incorporating a temporal aspect. Similar to Text-to-Image, concerns in SV include image quality and text-image relevance. However, the narrative aspect of SV implies that objects appearing in a visual frame must maintain

---

[1] Code: https://github.com/chrispapa2000/MaskGST

a consistent appearance in later frames. The most prominent examples of such objects are the main characters in the stories.

To address these challenges we construct a framework based on a MaskGIT [3] model. This approach has recently shown competitive results in Image Generation, both in terms of diversity and fidelity, whilst being significantly more efficient compared to auto-regressive Transformers and Diffusers [2, 3]. We enhance it with Cross-Attention sub-layers, to allow for past and future captions in the story to serve as context when generating an image. Cross-Attention mechanisms have been extensively adopted as a way to inform an input according to some context of different modality or origin [17, 20, 34]. Additionally, we employ a simple LLM-based caption augmentation technique to improve our model's robustness and attention on important textual concepts, similarly to [7] that applies such a method on a contrastive language-image task. Finally, we introduce a novel Character-Guidance technique, to prompt our model towards the generation of desired characters. Specifically, at inference, we form text-conditional logits, as well as logits conditioned on desired characters and logits conditioned on undesired characters. We combine those using a mechanism similar to the negative prompting formula from MUSE [2], to push text-conditional logits, towards the direction of the desired characters and away from the undesired ones. Our contributions can be summarized as follows:

- We are the first to employ the promising MaskGIT to construct a MaskGIT-style Transformer enhanced with Cross-Attention sub-layers for SV.
- We successfully employ an LLM for augmentation of the textual training data in a completely image-agnostic manner, achieving advanced robustness.
- We propose a simple, yet effective Character Guidance technique that significantly improves the generation of characters.
- We achieve SOTA results in terms of *Char-F1*, *Char-Acc* and *BLEU-2/3* metrics on Pororo-SV. Additionally, our model has the best *FID* score, compared to all previous Transformer-based SV architectures. Our results are strongly supported by a human survey.
- Our method is significantly more time-efficient than previous SOTAs, developed under tight resource constraints (16GB of vRAM).

2   Related Work

2.1   Text-to-Image Generation

The field of Text-to-Image Generation had been previously dominated by GANs [9, 32, 33]. However, GANs were notoriously unstable during training and usually specialized in a narrow space of visual themes. More recently, Diffusion Models [19, 21] have shown exceptional results in the task, by improving the quality of the generated images, whilst being able to generate a broader range of themes. However this comes at a cost of larger parameter counts, as well as increased training/inference times, imposing the need for high-end hardware and budget. At the same time, there have been several works that employ Transformers for

image generation. At first, auto-regressive Transformers [6, 18], that infer visual tokens in the latent space of a VQ-VAE [28] one by one were employed. More recently, Chang *et al*. [3] proposed MaskGIT, a parallel Transformer for image generation, while MUSE [2] built upon this idea using a two-stage approach, which achieves results comparable to those of Diffusion Models, whilst being significantly faster, even at a similar parameter size. However, the MaskGIT architecture still remains relatively under-explored, especially in the open-source realm, compared to Diffusers, that have been widely studied and adopted.

2.2   Story Visualization

Initial research in the field revolved around GANs, commencing with Story-GAN [11]. Several works build on top of this pivotal approach, including CP-CSV [25], DUCO-StoryGAN [13] and VLC-StoryGAN [12]. In terms of Transformers, two approaches have been published. VP-CSV [4] employs a two-stage approach, that starts by predicting the visual tokens in the image regions that correspond to characters and completes the background of the images at the second stage. CMOTA [1] uses memory modules to improve consistency and proposes a bidirectional approach (both text-to-image and image-to-text) to perform online caption augmentation, during training. Other than that, there has been a collection of recent diffusion-based approaches [8, 15, 24], all of which modify and fine-tune pre-trained LDM [19] for the task of SV. Finally, StoryLDM [17] and StoryGPT-V [19] leverage pre-trained LDM [19], as well. However, they focus on a modified version of SV where repeated character references in captions are replaced by pronouns (*e.g*. he, she, they).

3   Background

3.1   MaskGIT

Image Tokenization MaskGIT [3] works in the latent space of a VQ-GAN [6]. VQ-GAN's operation on an image $x \in \mathbb{R}^{3 \times H \times H}$ is summarized as follows:

$$z = \mathcal{Q}(\mathcal{E}(x)) \in \mathbb{R}^{D \times \frac{H}{f} \times \frac{H}{f}}, \ \hat{x} = \mathcal{D}(z) \in \mathbb{R}^{3 \times H \times H} \tag{1}$$

where $\mathcal{E}, \mathcal{Q}, \mathcal{D}$ are the Encoder, Quantizer and Decoder, respectively. The latent representation z is discrete, meaning that each D-dimensional vector in the ($\frac{H}{f} \times \frac{H}{f}$)-sized output of the Encoder is substituted via nearest-neighbor lookup, using a library of K visual tokens (this operation is carried out by $\mathcal{Q}$). H is the dimension of the images and f is a compression factor.

Text-to-Image Generation A Transformer is trained to predict the visual tokens of an image by conditioning on text tokens. The training and inference techniques proposed in [3] are summarized below.

Training An image is encoded into its discrete, latent representation z using VQ-GAN's Encoder and Quantizer. $Y = [y_i]_{i=1}^N$, with $N = (\frac{H}{f})^2$, is formed by flattening the visual tokens, z into a vector. $M = [m_i]_{i=1}^N$ is a random binary mask for all tokens. At each training step, the tokens in positions (i), where $m_i = 1$ are masked out (replaced by a special token, [MASK]). The Mask M is applied over Y to obtain $Y_{\bar{M}}$. MaskGIT's training objective is:

$$\mathcal{L}_{mask} = -\mathbb{E}[\sum_{\forall i \in [1,N], m_i=1} \log p_\theta(y_i | Y_{\bar{M}})] \tag{2}$$

where θ represents the parameters of the Transformer. Predicting randomly masked tokens during training allows for the utilization of an efficient inference technique (outlined in the next paragraph), so that multiple visual tokens to be predicted at each step, in contrast with traditional Next Token Prediction.

Inference Inference takes place over T iterations. It starts with all visual tokens masked out; $Y_{\bar{M}}^{(0)} = [[\texttt{MASK}]]_{i=1}^N$. In the t-th iteration, the masked tokens $(Y_{\bar{M}}^{(t)})$ are passed through the Transformer to predict probabilities $p^{(t)} \in \mathbb{R}^{N \times K}$ for all masked tokens. In every position, a token is sampled:

$$y_i^{(t)} \sim p_i^{(t)}, \ p_i^{(t)} \in \mathbb{R}^K, \ \forall i \in [1, N] \tag{3}$$

The probabilities according to which the tokens were sampled are now dealt with as confidence scores. The most confident tokens are kept unmasked and the rest of them are re-masked, to be predicted in a future iteration.

3.2   Caption Augmentation using LLMs

The outstanding capabilities of LLMs have been previously leveraged to perform text augmentation in the context of various tasks [5, 7, 27, 30, 31]. [7] proposes a method for augmenting captions of text-image pairs that are used to train a CLIP [16] model. At first, alternative captions are generated for a small number of text-image pairs, through various methods, including human annotation and chatbots. Original and generated captions are paired to form meta-input-output pairs. Subsequently, LLaMA [26] is used to produce alternative captions for all samples in the training data. The meta-input-output pairs are used as context for the LLM to better understand the task.

4   Method

4.1   Preliminaries

Let $S = \{s_1, ..., s_n\}$ and $X = \{x_1, ..., x_n\}$ be the story captions and story images respectively, where n is the number of caption-image pairs in a story. Additionally, we denote $z = \mathcal{Q}(\mathcal{E}(x))$ the latent, discrete representation of image x, encoded by VQ-GAN.

4.2 MaskGST

We propose MaskGST (Masked Generative Story Transformer), based on a modified version of MaskGIT [3] for SV. Following MaskGIT, we adopt VQ-GAN [6] for image tokenization. VQ-GAN's function is briefly discussed in Sec. 3.1.

Transformer MaskGST's Transformer (Fig. 1) employs two kinds of layers. *Full-Layers* have a Self-Attention sub-layer, followed by a Cross-Attention sub-layer, followed by a Feed-Forward Network (as in the Decoder of the original Transformer [29]). *Self-Layers* have the same structure, except that they omit Cross-Attention (as in the Encoder of the original Transformer [29]). The Transformer comprises of two Full-Layers, followed by four Self-Layers.

Input For the image-caption pair (x_i, s_i) the Transformer's input I_i is composed as follows:

$$I_i = (z_i^{FLAT}; s_i^{BPE}) \in \mathbb{R}^{(\frac{H}{f} \cdot \frac{H}{f} + L) \times d} \tag{4}$$

where $z_i^{FLAT} \in \mathbb{R}^{(\frac{H}{f} \cdot \frac{H}{f}) \times d}$ is the flattened version of $z_i = \mathcal{Q}(\mathcal{E}(x_i))$, projected to the Transformer's hidden space. $s_i^{BPE} \in \mathbb{R}^{L \times d}$ are the text embeddings that correspond to the BPE-encoding [22] of text caption s_i, with L being the number of text tokens in the representation. Finally, d is the hidden dimension of the Transformer.

Context The context utilized in the Cross-Attention sub-layers, for the i-th image of a story is:

$$ctx_i = (s_1^{BPE}; s_2^{BPE}; ...; s_n^{BPE}) \in \mathbb{R}^{(n \cdot L) \times d} \tag{5}$$

That is, when generating the i-th image, the model performs Cross-Attention to all captions in the story, both past and future. This way, in the first two Full-Layers of the generative process, the model is able to integrate relevant information from previous and consecutive captions into the visual and textual tokens of the input sequence. Then, in the following Self-Layers, the visual tokens of the input sequence are forged through mutual self-attention, as well as self-attention applied between them and the textual tokens of the input sequence, to form the final output sequence.

Training For Training we adopt MaskGIT's Masked Visual Token Modeling algorithm [3] (Sec. 3.1). For each training sample, a random subset of visual tokens is masked out in the input according to a binary mask M. The Transformer's final hidden states are projected to the K-dimensional space to form:

$$O_i = (O_i^{vis}; O_i^{text}) \in \mathbb{R}^{(\frac{H}{f} \cdot \frac{H}{f} + L) \times K}, \text{where } O_i^{vis} \in \mathbb{R}^{(\frac{H}{f} \cdot \frac{H}{f}) \times K} \tag{6}$$

The cross-entropy between the masked positions in O_i^{vis} and the corresponding ground-truth tokens in the VQ-GAN encoding z is calculated.

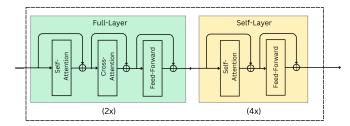*Inference* As in training, we adopt MaskGIT's iterative inference algorithm.

Fig. 1: MaskGST's Transformer model

4.3   Caption Augmentation Using an LLM

We use ChatGPT to augment the training captions, similarly to [7]. Instead of providing pairs of original and generated captions as context, we provide ChatGPT a description of its role as a caption augmentation assistant, along with information regarding the main characters in the dataset. Then, we prompt it with the captions of a story $S = \{s_1, ..., s_n\}$ in the following form:

$$1.\{s_1\} \; 2.\{s_2\} \; ... \; n.\{s_n\} \tag{7}$$

ChatGPT returns an alternative version for each one of the captions:

$$1.\{\hat{s}_1\} \; 2.\{\hat{s}_2\} \; ... \; n.\{\hat{s}_n\} \tag{8}$$

At training time, we randomly pick either the original or the generated caption for each training sample at each epoch. Using different text descriptions for the same image is expected to shield the generative process against over-fitting and help the Transformer distinguish between important and insignificant textual concepts when generating visual tokens. At inference, we only use the original captions of the dataset. Notably, this is a completely image-agnostic caption augmentation method, *i.e.* the LLM provides alternative captions without access to the image it is describing. In contrast to [7], where the augmented data are used to train a contrastive model, we use the alternative captions to train a generative model, where the accuracy of the descriptions is of greater importance. An example of a caption augmentation prompt is provided in Appendix A.

4.4   Character Guidance

We propose Character Guidance, an auxiliary method to improve the generation of characters in the images. An additional library of $2n_c$ Character Embeddings (CE) is introduced to the Transformer, where n_c is the number of main, recurring characters in the dataset. For each character, we have a positive embedding (the character is referenced in the caption) and a negative one (the character is not referenced in the caption). When using this technique, we concatenate n_c embeddings to the input of the Transformer, one for each character (positive for characters that are referenced in the description and negative for the rest). The input now becomes:

$$I_i = (z_i^{FLAT}; s_i^{BPE}; emb_i^c) \in \mathbb{R}^{(\frac{H}{f} \cdot \frac{H}{f} + L + n_c) \times d} \tag{9}$$

where $emb_i^c \in \mathbb{R}^{n_c \times d}$ is the concatenation of positive CE, for characters that are *present* in the current caption (s_i) and negative CE for characters that are *absent* from the caption.

Training In order to reinforce the model's focus on the CE, we completely drop the text conditioning for a percentage of training samples per batch, by replacing all text embeddings with a [NULL] embedding and keep only the CE. Other than that, the training process remains intact.

Inference During inference, we compute three sets of logits[2] when generating an image, corresponding to the prompts described below.

Text-Conditional Prompt The first set of logits ($\ell_{tc} \in \mathbb{R}^{N \times K}$)[3] is computed by conditioning the generative process on text descriptions.

Positive Character Prompt The second set of logits ($\ell_{char} \in \mathbb{R}^{N \times K}$) is computed using only Character Embeddings and completely dropping the captions.

Negative Character Prompt We compute a third set of logits $\ell_{\overline{char}}$. To compute $\ell_{\overline{char}}$ we completely drop text descriptions from the transformer's input, as we do when computing ℓ_{char}. However, instead of using positive embeddings for characters present in the description and negative ones for absent characters, we do the opposite. Negative embeddings are used for characters *present* in the description and positive embeddings for characters *absent* in the description. In a sense, $p_{\overline{char}}$ is computed using the "*complement*" of the input that is used to generate p_{char}. We employ Negative Prompting, inspired by the way it is used in [2]. However, [2] uses this technique to push the logits away from a negative text prompt, that needs to be provided by the user, while, we perform Negative Prompting using a character prompt, that can be automatically formed, by observing which characters are absent from the given text prompt.

Computing the Final Logits The final logits are now computed as follows:

$$\ell = (1 - \lambda)\ell_{tc} + 2\lambda\ell_{char} - \lambda\ell_{\overline{char}}, \ \lambda \in [0, 1) \tag{10}$$

This combination of logits is formed at every step of the iterative inference process. Softmax is applied to obtain probabilities at each one of the N positions. These probabilities are used to sample tokens Y. ℓ_{tc} encode the specific information regarding the generation of an image from its description, while ℓ_{char} encapsulate information solely regarding the presence of desired characters in

---

[2] The term *logits* refers to the unnormalized outputs of the Transformer

[3] $N = (\frac{H}{f})^2$, while K is the size of the VQ-GAN's token library

| Pororo | Crong | Eddy | Harry | Loopy | Petty | Poby | Rody | Tongtong |

Fig. 2: Main characters featured in Pororo-SV

the image. Similarly, $\ell_{\overline{char}}$ encode information about undesired characters. By adding ℓ_{char} and subtracting $\ell_{\overline{char}}$ from ℓ_{tc} we attempt to actively push the text-conditional logits towards the generation of the desired characters and away from the generation of undesired characters.

5   Experiments and results

5.1   Experimental Setup

Software and Hardware Our entire codebase is developed in PyTorch. For image tokenization we use VQ-GAN's [6] original implementation[4]. Our Transformer model is adapted from an open-source implementation of MUSE [2].[5] We perform all our experiments - both training and inference - on a single NVIDIA V100 GPU (16GB). We consider this to be a strong indicator for the resource-friendliness of our approach.

Dataset We train and test our approach on Pororo-SV [11], which is the most widely adopted benchmark in previous SV works. We adopt the split proposed in [13], which comprises of 10191/2334/2208 train/validate/test stories. Each story contains 5 images (64×64 pixels) and the corresponding captions. There are 9 recurring characters in the dataset, shown in Fig. 2. We choose Pororo-SV because there are publicly available evaluation models for it, that have been widely adopted in the past and make it easier to compare our results with previous baselines.

Evaluation metrics Following previous works, we adopt *FID* to assess image quality. We also employ *Char-F1* and *Char-Acc*, proposed in [13], to evaluate character generation. Finally, we report BLEU scores based on video redescription [13] to evaluate global semantic alignment. More details on the metrics are given in Appendix E.

Baselines We compare our approach with several previous arts, using the reported results of the respective papers. Among GANs we only compare it with VLC-StoryGAN [12], since it has been shown to be the best of them. In terms of Transformers, there are two baselines: VP-CSV [4] and CMOTA [1].

---

[4] https://github.com/CompVis/taming-transformers
[5] https://github.com/lucidrains/muse-maskgit-pytorch

It would be unfair to include recent diffusion-based approaches [8, 15, 24] in our main comparison, since they leverage pre-trained Latent Diffusion Models (LDM) [19], trained on a vast dataset with massive compute. They also require excessively expensive hardware to run on. For reference, AR-LDM [15] reports using 8 NVIDIA A100 (80GB), which is equivalent to [40×] the vRAM we use. Besides, they only report FID scores, which we deem insufficient for SV on their own. However, we acknowledge that in terms of FID they are superior to our model. We elaborate more on this in Appendix C. [17] and [23] are applied on a modified version of the task, therefore a direct comparison is not applicable.

5.2 Experiments

All of our models are trained from scratch. First, we train a VQ-GAN, with a downsampling factor $f = 8$, which corresponds to a 8×8 resolution for the latent image representations (since the image resolution is 64×64). We use a library of $K = 128$ latent embeddings. After completing VQ-GAN's training, we train MaskGST's Transformer. The Transformer has 6 layers; 2 full layers, followed by 4 Self-Layers, as shown in Fig. 1. We use Scaled Dot-Product Attention [29], with the number of heads set to $n_{head} = 8$. Based on the results reported in [3], a cosine function is adopted for mask scheduling. We train for 250 Epochs, with a learning rate $lr = 1e - 3$. Caption Augmentation (Sec. 4.3) and Character Guidance (Sec. 4.4) are employed during training. To reinforce Character Guidance, we drop the text descriptions in 20% of training samples, at each Epoch. Two variants of the Transformer are trained; one with hidden dimension $d = 1024$ and one with $d = 2048$. At inference we use both Positive and Negative Prompting, in terms of Character Guidance. To that end, we form logits as shown in Eq. (10), with $\lambda = 0.2$. We perform inference over $T = 20$ timesteps, for all our models.

5.3 Quantitative Results

We gather our results, as well as results from previous works in Tab. 1. In the names of our models $CG_{\pm}$ refers to *Character Guidance (positive and negative)*, while *w/ aug. captions* is used for models trained with caption augmentation.

MaskGST-CG$_{\pm}$ w/ aug. captions ($d = 1024$) Our model with $d = 1024$ performs better than all previous GAN and Transformer architectures, in all metrics (lowest FID and highest Char-F1, Char-Acc, BLEU-2/3). Especially Character metrics are raised significantly. Specifically, in Char-F1 there is a 3.6 point improvement, while in Char-Acc we achieve an improvement of 7.7 points, compared to the previous best (VP-CSV). We largely attribute our models superiority in terms of character generation, to our Character Guidance mechanism.

MaskGST-CG$_{\pm}$ w/ aug. captions ($d = 2048$) Doubling the hidden dimension to $d = 2048$ improves our results across all metrics. Most notably, there is a

Table 1: Results from our models ($d = 1024$ and $d = 2048$) and previous baselines, for the test set of Pororo-SV. We report scores for *FID* (lower is better), as well as *Char-F1*, *Char-Acc*, *BLEU-2/3* (higher is better).

Model Family	Model	FID($\downarrow$)	Char-F1	Char-Acc	BLEU-2/3($\uparrow$)
GAN	VLC-StoryGAN [12]	84.96	43.02	17.36	3.80/1.44
Auto-Regressive	VP-CSV [4]	65.51	56.84	25.87	4.45/1.80
Transformer	CMOTA [1]	52.13	53.25	24.72	4.58/1.90
MaskGST-CG$_\pm$	d=1024	51.65	60.46	33.62	4.82/2.00
w/ aug. captions	d=2048	**42.86**	**65.10**	**37.50**	**5.13/2.26**

major improvement in terms of FID, which decreases by 8.8 points compared to the $d = 1024$ version of our model. We assume that doubling the hidden dimension leaves more room to the model to learn more complex and fine mappings between words and visual features, which result in higher quality images with more details. The model's ability to learn more complex representations and produce more detailed images can account for the improvements in the other metrics as well. Specifically, high quality images will depict improved versions of the characters and produce better BLEU scores through video captioning. In terms of Char-F1, Char-Acc and BLEU-2/3, this model achieves the current SOTA, compared to all previous methods.

5.4   Qualitative Results

In Fig. 3 we provide four examples of image-stories from the Pororo-SV test set. For each story, we provide its captions, the ground-truth images (Original), the image sequence generated by CMOTA [1] and the one generated by our best model (MaskGST-CG$_\pm$ w/ aug. captions ($d = 2048$)). To that end, we used the pre-trained CMOTA model that was released by its authors[6]. CMOTA is the most recent Transformer model applied to the task of SV. We carry out a qualitative comparison based on these examples.

Images generated by our model are of higher **Visual Quality**, with clearer and more accurate features, compared to CMOTA's that tend to be blurry. In terms of **Temporal Consistency**, our model performs better as well. This is showcased by the fact that backgrounds and characters are well maintained across the sequences, contrary to CMOTA that is prone to switching between different backgrounds (indoor and outdoor). Concerning **Semantic Relevance** (caption-image alignment) our model is again superior. This is evident by the fact that it generates the exact set of characters mentioned in captions, with remarkable accuracy. On the contrary CMOTA struggles, especially in instances where multiple characters are referenced. A more detailed qualitative comparison is provided in Appendix B.

---

[6] https://github.com/yonseivnl/cmota

Frame 1: Crong eats meat. Loopy gave a dish of vegetables.
Frame 2: Crong doesn't like vegetables. Crong pushed the dish of vegetables.
Frame 3: Both Pororo and Crong look at the meat in the dish.
Frame 4: Crong picked up the last meat.
Frame 5: Pororo told that Crong is so greedy.

Frame 1: Loopy is back with Poby.
Frame 2: Loopy and Poby notice that chair is not broken.
Frame 3: Poby and Loopy thinks the chair is very strange.
Frame 4: Loopy is looking at chair with question.
Frame 5: Loopy thanks Poby for coming.

Frame 1: Petty asks if Crong really okay is.
Frame 2: Loopy says Crong looks not that good.
Frame 3: Crong trying to pretend to be okay waves Crong head.
Frame 4: Crong pretends to do some freehand exercise.
Frame 5: Pororo laughs at Crong calling Crong as a regular pooping machine.

Frame 1: Loopy talks and spreads Petty arms. Petty looks at Loopy smiles and nods.
Frame 2: Poby gathers red car hands and talks. Loopy Petty and Harry are looking at Poby.
Frame 3: Loopy Poby and Petty walk stop and turn back.
Frame 4: Harry looks angry and talks.
Frame 5: Harry looks at Poby and turns red car head.

Fig. 3: Qualitative Comparison between our model (MaskGST-CG$_\pm$ w/ aug. captions) and CMOTA [1] across 4 story examples.

5.5   Human Evaluation

In order to further investigate the qualitative results of Sec. 5.4, we conducted a human survey across these three criteria which were also adopted by previous works [1, 12, 13], comparing our model with CMOTA [1]. The evaluation is done over 100 stories from the test set of Pororo-SV. Each story is evaluated by 2 distinct annotators. The results of the study (Tab. 2) indicate that our model is superior across all 3 criteria, thus supporting our quantitative results. More details on the human study can be found in the Appendix F.

5.6   Ablation Study

We perform an ablation study in order to determine the specific effects of the different components of our approach. We consider MaskGST (Sec. 4.2) to be our

Table 2: Results of our human survey. We compare the results of our model *MaskGST-CG$_\pm$ w/ aug. caption (d=2048)* (Ours) against CMOTA's, across three criteria. Our(%) and CMOTA(%) indicate the percentage of cases where each model was chosen by both annotators, whereas Tie(%) accounts for the remaining cases.

Criterion	Ours (%)	CMOTA (%)	Tie(%)
Visual Quality	**78%**	3%	19%
Temporal Consistency	**66%**	8%	26%
Semantic Relevance	**64%**	9%	27%

basic approach. Using *Caption Augmentation* (Sec. 4.3) and employing *Character Guidance* (Sec. 4.4) are the two major components that build on top of it, which we evaluate separately. Tab. 3 shows the results for these experiments, on the test set of Pororo-SV. We use Transformers with $d = 1024$.

Caption Augmentation We observe that *Caption Augmentation* (Tab. 3, MaskGST w/ aug. captions) brings substantial imporovements across all metrics, with the exception of BLEU scores, that are slightly lower, compared to the baseline. We believe that providing alternative captions for the same image, during training is beneficial for two reasons. On the one hand, it shields the model against over-fitting. Additionally, it guides the model to focus on more important textual concepts (*e.g.* character names), when generating the visual tokens, since such concepts will probably appear in both versions of a caption.

Character Guidance *Character Guidance* (Positive and Negative) (Tab. 3, MaskGST-CG$_\pm$) has a dramatic effect on all metrics, compared to the baseline. This observation confirms that Character Guidance indeed succeeds in directly impacting the generation of characters by providing separate logits that guide the model towards the production of the correct subset of them. This is evident by the increase in Char-F1 and Char-Acc, by 9 and 7.5 points each. The improvement in character generation enhances the overall quality of the images, which is reflected in the additional improvements in FID and BLEU scores.

Caption Augmentation & Character Guidance Combining the two methods (Tab. 3, MaskGST-CG$_\pm$ w/ aug. captions) seems to be beneficial. FID and Char-F1 are further improved, with Char-Acc staying the same, as in MaskGST-CG$_\pm$, while BLEU-2/3 are slightly lower, yet still improved compared to the baseline.

5.7   Study on Character Guidance Factor (λ)

Table 3: Ablation Study for our method *MaskGST-CG$_\pm$ w/ aug. captions*. We conduct experiments with hidden dimension $d = 1024$. We report scores for *FID* (lower is better), as well as *Char-F1*, *Char-Acc*, *BLEU-2/3* (higher is better).

Component	Model	FID($\downarrow$)	Char-F1	Char-Acc	BLEU-2/3($\uparrow$)
baseline	MaskGST	66.12	50.48	26.12	4.68/2.01
+ aug. captions	{MaskGST w/ aug. captions}	59.91	54.64	28.67	4.45/1.81
+ char. guid.	MaskGST-CG$_\pm$	54.95	59.55	33.64	4.96/2.10
+{aug. captions, char. guid.}	{MaskGST-CG$_\pm$ w/ aug. captions}	51.65	60.46	33.62	4.82/2.00

Using our best model, *MaskGST-CG$_\pm$ w/ aug.captions* (d=2048), we perform an analysis on the Character Guidance factor λ (Eq. (10)). Our results are summarized in Fig. 4. Character metric curves increase with the increase in λ (with a slight decrease for $\lambda = 0.8$). This is to be expected, since by increasing λ we pay more attention to the character logits vs the text-conditional logits. FID, on other hand decreases (improves) until $\lambda = 0.4$ and then it increases for the next two experiments. This is explained by the fact that improving the generation of Characters, also improves the overall quality of the images, to a certain extend, since Characters are a significant part of them. Once λ becomes too large,

Fig. 4: Comparison of Character Guidance Factor (λ) values for different evaluation metrics.

there is an adverse effect on overall image quality, since aspects of images other that characters start to be neglected, because character logits are weighed disproportionately highly. For $\lambda = 0.4$ we get the optimal combination of metrics (68.32, 41.40 and 42.49 for Char-F1, Char-Acc and FID, respectively). However, in practice, we find that even $\lambda = 0.4$ makes the model focus too much on character generation and has an adverse effect on inter-image coherence. We empirically find $\lambda = 0.2$ to achieve the best qualitative trade-off between Character Generation and overall quality of image-stories. We elaborate more on our empirical findings in Appendix D.

5.8   Resource Usage Analysis and Training/Inference Times

We compare our best model with previous Transformer architectures in terms of training resources as well as Training/Inference Times. In terms of Training, we compare with VP-CSV [4], that reports such information (Tab. 4). CMOTA [1]

Table 4: Training times/resources comparison between our models (d=1024 and d=2048) with VP-CSV. We use the stats reported by the authors of VP-CSV [4].

Metric	Ours (d=1024)	Ours (d=2048)	VP-CSV
Available GPUs	[1×] V100 (16GB)	[1×] V100 (16GB)	[4×] A100 (40GB)
GPU usage	576 (GB · hours)	1712 (GB · hours)	1920 (GB · hours)

Table 5: Comparison of inference times between our models (d=1024 and d=2048) with CMOTA [1]. We compare based on inference runs on the same NVIDIA V100 (16GB).

Metric	Ours (d=1024)	Ours (d=2048)	CMOTA
GPU usage	9.07 (GB · hours)	25.07 (GB · hours)	60.8 (GB · hours)

does not specify this information, however since we perform inference with it, we can carry out a comparison with our model, in terms of inference times, on the same GPU (Tab. 5). We report GPU usage in (GB·hours), *i.e.* the GBs of vRAM in the GPU/GPUs, times the hours it was in use. It is evident that both of our models are significantly more efficient compared to previous Transformers applied to the task, while achieving better performance across all metrics. More details on our calculations can be found in the Appendix G.

6   Limitations and Impacts

We acknowledge that our models are only evaluated on a Cartoon dataset, which may be limiting for real-world applications. Additionally, despite our models' merit, they still suffer in terms of FID, where large pre-trained models [8, 15, 24] remain unparalleled. Regarding the impact of our models, we cannot foresee any direct missuse of them, since they are trained to produce cartoons. Having said that, we are strongly opposed to any negative use of our work, that hurts individuals or violates community guidelines and best practices in any way.

7   Conclusion

In this paper, we adopt a MaskGIT model and enhance it with Cross-Attention sub-layers for the task of Story Visualization. In addition, we propose a novel Character Guidance method that improves the generation of characters, by combining text-conditional and character-conditional outputs in the logit space. We also employ an image-agnostic, LLM-driven caption augmentation technique and show that it can be successfully used for generative tasks. Using this combined approach, we achieved SOTA results over multiple metrics on Pororo-SV, under a tight computational budget. We believe that our work should encourage further research in the field of MaskGIT architectures, for generative vision tasks.

On the other hand, our Character Guidance method could be explored more extensively in the context of SV and be paired with large, possibly pre-trained models. Otherwise, it could be extended to other generative tasks, where there is a particular interest for a specific set of concepts, like characters in SV.
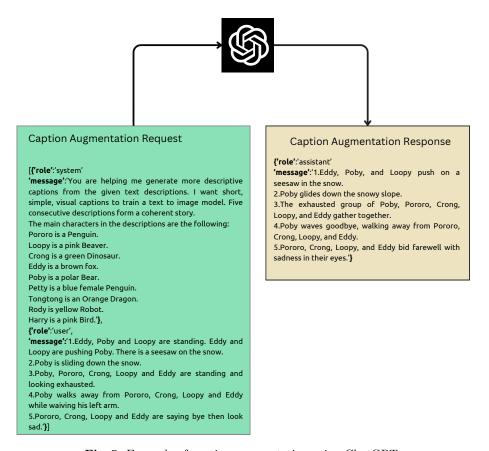
References

1. Ahn, D., Kim, D., Song, G., Kim, S.H., Lee, H., Kang, D., Choi, J.: Story visualization by online text augmentation with context memory. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3125–3135 (2023)
2. Chang, H., Zhang, H., Barber, J., Maschinot, A., Lezama, J., Jiang, L., Yang, M.H., Murphy, K., Freeman, W.T., Rubinstein, M., Li, Y., Krishnan, D.: Muse: Text-to-image generation via masked generative transformers (2023)
3. Chang, H., Zhang, H., Jiang, L., Liu, C., Freeman, W.T.: Maskgit: Masked generative image transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11315–11325 (2022)
4. Chen, H., Han, R., Wu, T.L., Nakayama, H., Peng, N.: Character-centric story visualization via visual planning and token alignment. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp. 8259–8272 (2022)
5. Dai, H., Liu, Z., Liao, W., Huang, X., Cao, Y., Wu, Z., Zhao, L., Xu, S., Liu, W., Liu, N., Li, S., Zhu, D., Cai, H., Sun, L., Li, Q., Shen, D., Liu, T., Li, X.: Auggpt: Leveraging chatgpt for text data augmentation (2023)
6. Esser, P., Rombach, R., Ommer, B.: Taming transformers for high-resolution image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12873–12883 (2021)
7. Fan, L., Krishnan, D., Isola, P., Katabi, D., Tian, Y.: Improving clip training with language rewrites. Advances in Neural Information Processing Systems **36** (2024)
8. Feng, Z., Ren, Y., Yu, X., Feng, X., Tang, D., Shi, S., Qin, B.: Improved visual story generation with adaptive context modeling. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Findings of the Association for Computational Linguistics: ACL 2023. pp. 4939–4955. Association for Computational Linguistics, Toronto, Canada (Jul 2023). `https://doi.org/10.18653/v1/2023.findings-acl.305`, `https://aclanthology.org/2023.findings-acl.305`
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
10. Li, J., Li, D., Xiong, C., Hoi, S.: Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In: International Conference on Machine Learning. pp. 12888–12900. PMLR (2022)
11. Li, Y., Gan, Z., Shen, Y., Liu, J., Cheng, Y., Wu, Y., Carin, L., Carlson, D.E., Gao, J.: Storygan: A sequential conditional gan for story visualization. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6322–6331 (2018), `https://api.semanticscholar.org/CorpusID:54457433`
12. Maharana, A., Bansal, M.: Integrating visuospatial, linguistic, and commonsense structure into story visualization. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. pp. 6772–6786 (2021)
13. Maharana, A., Hannan, D., Bansal, M.: Improving generation and evaluation of visual stories via semantic consistency. In: Proceedings of the 2021 Conference of

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 2427–2442 (2021)

14. Maharana, A., Hannan, D., Bansal, M.: Storydall-e: Adapting pretrained text-to-image transformers for story continuation (2022)

15. Pan, X., Qin, P., Li, Y., Xue, H., Chen, W.: Synthesizing coherent story with auto-regressive latent diffusion models. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2920–2930 (2024)

16. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)

17. Rahman, T., Lee, H.Y., Ren, J., Tulyakov, S., Mahajan, S., Sigal, L.: Make-a-story: Visual memory conditioned consistent story generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2493–2502 (2023)

18. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In: International Conference on Machine Learning. pp. 8821–8831. PMLR (2021)

19. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)

20. Ruan, L., Ma, Y., Yang, H., He, H., Liu, B., Fu, J., Yuan, N.J., Jin, Q., Guo, B.: Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10219–10228 (2023). https://doi.org/10.1109/CVPR52729.2023.00985

21. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in Neural Information Processing Systems **35**, 36479–36494 (2022)

22. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units (2016)

23. Shen, X., Elhoseiny, M.: Storygpt-v: Large language models as consistent story visualizers (2023)

24. Song, T., Cao, J., Wang, K., Liu, B., Zhang, X.: Causal-story: Local causal attention utilizing parameter-efficient tuning for visual story synthesis (2023)

25. Song, Y.Z., Tam, Z.R., Chen, H.J., Lu, H.H., Shuai, H.H.: Character-preserving coherent story visualization. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)

26. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G.: Llama: Open and efficient foundation language models (2023)

27. Ubani, S., Polat, S.O., Nielsen, R.: Zeroshotdataaug: Generating and augmenting training data with chatgpt (2023)

28. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)

29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6000–6010. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)

30. Whitehouse, C., Choudhury, M., Aji, A.F.: Llm-powered data augmentation for enhanced cross-lingual performance (2023)
31. Wozniak, S., Kocon, J.: From big to small without losing it all: Text augmentation with chatgpt for efficient sentiment analysis. In: 2023 IEEE International Conference on Data Mining Workshops (ICDMW). pp. 799–808. IEEE Computer Society, Los Alamitos, CA, USA (dec 2023). `https://doi.org/10.1109/ICDMW60847.2023.00108`, `https://doi.ieeecomputersociety.org/10.1109/ICDMW60847.2023.00108`
32. Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., He, X.: Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1316–1324 (2018)
33. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In: Proceedings of the IEEE international conference on computer vision. pp. 5907–5915 (2017)
34. Zheng, J., Liu, H., Feng, Y., Xu, J., Zhao, L.: Casf-net: Cross-attention and cross-scale fusion network for medical image segmentation. Comput. Methods Prog. Biomed. **229**(C) (feb 2023). `https://doi.org/10.1016/j.cmpb.2022.107307`, `https://doi.org/10.1016/j.cmpb.2022.107307`

A   Details on Caption Augmentation via ChatGPT

For every five-caption story in the training set, we carry out a caption augmentation conversation with ChatGPT. The conversation starts with a *system message* that describes its role as a caption augmentation assistant. Additionally, this message contains basic information about the characters in the dataset. We presume that the characters in Pororo-SV constitute a very specific domain of knowledge. Therefore we consider this information to be necessary domain-specific context for the model to provide suitable alternative captions. The next message, in the conversation contains the captions of a story, numbered from 1 to 5. ChatGPT responds with a message containing the alternative captions, numbered in the same way. We use ChatGPT3.5 Turbo, through the API. An example can be seen in Figure 5.

Caption Augmentation Request

[{**'role'**:'system'
'message':'You are helping me generate more descriptive captions from the given text descriptions. I want short, simple, visual captions to train a text to image model. Five consecutive descriptions form a coherent story.
The main characters in the descriptions are the following:
Pororo is a Penguin.
Loopy is a pink Beaver.
Crong is a green Dinosaur.
Eddy is a brown fox.
Poby is a polar Bear.
Petty is a blue female Penguin.
Tongtong is an Orange Dragon.
Rody is yellow Robot.
Harry is a pink Bird.'**},**
{**'role'**:'user',
'message':'1.Eddy, Poby and Loopy are standing. Eddy and Loopy are pushing Poby. There is a seesaw on the snow.
2.Poby is sliding down the snow.
3.Poby, Pororo, Crong, Loopy and Eddy are standing and looking exhausted.
4.Poby walks away from Pororo, Crong, Loopy and Eddy while waiving his left arm.
5.Pororo, Crong, Loopy and Eddy are saying bye then look sad.'**}]**

Caption Augmentation Response

{**'role'**:'assistant'
'message':'1.Eddy, Poby, and Loopy push on a seesaw in the snow.
2.Poby glides down the snowy slope.
3.The exhausted group of Poby, Pororo, Crong, Loopy, and Eddy gather together.
4.Poby waves goodbye, walking away from Pororo, Crong, Loopy, and Eddy.
5.Pororo, Crong, Loopy, and Eddy bid farewell with sadness in their eyes.'**}**

Fig. 5: Example of caption augmentation using ChatGPT.

B   Detailed Qualitative Comparison

In this section we provide a more thorough qualitative comparison with CMOTA [1], based on the examples that are provided in the main paper. The following comments refer to Fig. 3.

Image Quality In terms of image quality, it is evident from the examples that our model is superior to CMOTA. For example, in the top-left panel, all of our images contain discernible objects. Additionally, the characters are depicted with high quality (*e.g.* the dinosaur's eyes and the penguin's beak). On the contrary, most of CMOTA's images in the same example are blurry, while some of them are completely incomprehensible.

Temporal Consistency Our image-stories are more coherent than CMOTA's across all examples. Especially, in the top-right panel, our model achieves remarkable results with the room remaining similar in all images and the characters holding similar positions, as well.CMOTA, on the other hand, struggles to maintain these features, as evidenced by the contrast between the first and final images, which depict outdoor scenes, and the three middle ones, which are from an indoor space.

Semantic Relevance The term *Semantic Relevance* refers to whether the generated image is relevant to the corresponding caption. In regards to this, our model shows remarkable capabilities, especially in terms of producing the mentioned characters. For instance, in the bottom-right panel, it manages to produce the correct subset of characters in all images, although that changes constantly. On the contrary, CMOTA struggles to produce multiple characters. This is most evident in the second image of the botton-right example, where the attempt to produce a big number of characters collapses into a blurry result.

C   Comparison with Large Pre-Trained Models

As we have stated in the main paper, there has been a series of recent approaches [8,15,24] that modify and fine-tune Latent Diffusion Models (LDMs) [19] for the task of Story Visualization. A direct comparison with them is not fair, due to the extensive pre-training of LDM, as well as the significantly superior hardware that is used to develop these models, compared to ours. However, for the sake of completeness, we devote this section to a quantitative comparison.

AR-LDM [15] models the image sequence generation in an auto-regressive manner. Specifically, frames are generated by a Latent Diffusion Model [19] one by one, from the first one to the last one. The Diffusion process is conditioned on the current caption, as well as a multi-modal representation of each previous caption and generated image pair. The current caption is encoded using CLIP [16], while multi-modal features from previous caption-image pairs are extracted

via BLIP [10]. ACM-VSG [8] similarly models frame generation auto-regressively, whilst conditioning on past multi-modal context. However, it also introduces an adaptive guidance mechanism that aims to push frames of similar captions, to be similar as well. Causal-Story [24] improves AR-LDM by introducing a local, causal attention mask that limits the size of historical context tokens, to eliminate confusion, caused by interfering captions.

C.1   Story Visualization

In Table 6 we provide a comprehensive historical overview of results on the task of SV. We include large-scale diffusion models [8, 15, 24], along with all other previous works, as well as our models with $d = 1024$ and $d = 2048$. As it is evident from the table, all diffusion models are evaluated solely on FID, where they are by far better than all other approaches, including ours. Among other approaches, that are trained from scratch, ours holds the best FID, as well as the overall SOTA in terms of all other metrics.

Table 6: Story Visualization results for all previous works, including recent diffusion models, on the test set of Pororo-SV. We report scores for *FID* (lower is better), as well as *Char-F1*, *Char-Acc*, *BLEU-2/3* (higher is better). Parameter counts for [13], [12] and [1] are taken from [1]. Parameters counts for [15] and [8] are takens from [8].

Model Family	Model	#param	FID($\downarrow$)	Char-F1	Char-Acc	BLEU-2/3($\uparrow$)
GAN	StoryGAN [11]	-	158.06	18.59	9.34	3.24 / 1.22
	CP-CSV [25]	-	140.24	21.78	10.03	3.25 / 1.22
	DUCO-SG [13]	101M	96.51	38.01	13.97	3.68 / 1.34
	VLC-SG [12]	100M	84.96	43.02	17.36	3.80 / 1.44
Auto-Regressive Transformer	VP-CSV [4]	-	65.51	56.84	25.87	4.45 / 1.80
	CMOTA [1]	96.6M	52.13	53.25	24.72	4.58 / 1.90
MaskGST-CG$_\pm$ w/ aug. captions	d=1024	105M	51.65	60.46	33.62	4.82 / 2.00
	d=2048	276M	42.86	**65.10**	**37.50**	**5.13 / 2.26**
Diffusion Model	AR-LDM [15]	1.5B	16.59	-	-	-
	ACM-VSG [8]	850M	**15.48**	-	-	-
	Causal-Story [24]	-	16.28	-	-	-

C.2   Story Continuation

[8, 15, 24] also report results for the task of Story Continuation [14]. Story Continuation (SC) is similar to SV. However, in SC the first frame is considered to be given as input and the rest of the frames need to be generated. In Table 7 we report SC results for our model ($d = 2048$), as well as AR-LDM [15], ACM-VSG [8] and Causal-Story [24]. Additionally, we include results for StoryDALLE (fine-tune) [14] and Mega-StoryDALLE [14] that are both based on pre-trained auto-regressive transformers. These where the first models applied to the task of SC.

We observe that although our model has the lowest parameter count, by far, it outperforms all the other large, extensively pre-trained models in terms of Char-F1 and Char-Acc. This speaks of the remarkable merit of our Character Guidance method. In terms of FID, large pre-trained models remain unparalleled, as in SV. Since our model is trained for SV, in order to evaluate it for SC, we just discard the first generated frame and evaluate it using the remaining four frames.

Table 7: Story Continuation results on the test set of Pororo-SV, for large pre-trained models, as well as MaskGST-CG$_\pm$ w/ aug. captions (d=2048) . We report scores for *FID* (lower is better), as well as *Char-F1*, *Char-Acc* (higher is better).

Model	#param	FID($\downarrow$)	Char-F1	Char-Acc
StoryDALLE(fine-tune) [14]	1.3B	25.90	36.97	17.26
Mega-StoryDALLE [14]	2.8B	23.48	39.91	18.01
AR-LDM [15]	1.5B	17.40	-	-
ACM-VSG [8]	850M	**15.36**	45.71	22.62
Causal-Story [24]	-	16.98	-	-
Ours (d=2048)	276M	43.31	**65.32**	**37.38**

D   Empirical Results for the Character Guidance Factor (λ)

In the main paper, we present a study on the Character Guidance Factor (λ). As we point out there, despite the fact that $\lambda = 0.4$ yields better quantitative results, we empirically find that $\lambda = 0.2$ achieves the best trade-off between character generation and overall quality of generated stories. Specifically, for $\lambda = 0.4$, the model becomes more prone to inconsistency regarding the background of different story frames. In Figure 6 we provide three examples that demonstrate this observation. In all three image-stories generated with $\lambda = 0.4$, there is an interleaving of indoor and outdoor backgrounds, whereas for $\lambda = 0.2$, the background is held more consistent.

E   Details on Reported Metrics

Following previous works, we adopt *FID* to assess the quality of the generated images. In order to evaluate the presence of main characters in the images we use *Char-F1* and *Char-Acc*, proposed in [13]. These metrics are calculated using a multi-label classifier that recognizes the presence of character in the generated images. The classifier's predictions are compared to the ground-truth character references in the captions to calculate F1-score and Accuracy. We use the fine-tuned Inception-V3 from the original paper, available here[7]. Finally, we follow

---

[7] https://github.com/adymaharana/StoryViz

Frame 1: Eddy has a sullen face and Eddy apologizes to all the friends about his spying.
Frame 2: Eddy felt sorry and apologizes. all the friends are seeing Eddy.
Frame 3: Being angry Loopy confirms that Eddy should promise not to do it again.
Frame 4: Eddy promises not to observe his friends ever again.
Frame 5: Eddy says Eddy will show them something as his apology when it becomes night.

Frame 1: Poby says something to Pororo about a Crong's interest.
Frame 2: Pororo says something thinking about Crong.
Frame 3: Pororo imagines that Crong is drawing a plenty of pictures on the floor with a happy. Pororo cannot repress Pororo's astonishment
Frame 4: Pororo imagines that Crong is drawing a plenty of pictures on the floor with a happy. Pororo cannot repress Pororo's astonishment Pororo shakes his head passionately.
Frame 5: Pororo ran to his house with an anger face. Pororo's friends look at the Pororo's back.

Frame 1: Pororo fixes his snowman that wind has ruined. Pororo realizes that it was wind who ruined his snowman.
Frame 2: Pororo is standing next to his snowman. Pororo says to Eddy that Pororo doubted Eddy for ruining his snowman.
Frame 3: Eddy is standing next to his snowman. Eddy says sorry to Pororo for doubting Pororo. Pororo and Eddy looks at each other.
Frame 4: Pororo and Eddy cleared up their misunderstandings. the sky gets brighter. time has passed.
Frame 5: Pororo and Eddy's snowman are standing together smiling. Pororo and Eddy are smiling also.

Fig. 6: Three examples of generated image sequences that highlight the fact that a value of $\lambda = 0.4$ hinders the coherence between images, compared to the results for $\lambda = 0.2$.

[13] in using a video captioning model that produces a single caption for each generated image-story. The generated captions are compared to the ground truth ones to calculate *BLEU* scores. We use the fine-tuned video captioner, available at the same link as the classifier.

F   Details on Human Evaluation Survey

We conducted a human survey to evaluate the stories generated by our model vs CMOTA [1]. The evaluation is done across 100 image-stories. Each story is evaluated across three criteria:

- *Visual Quality* refers to whether the images are visually appealing, rather than blurry and difficult to understand.
- *Temporal Consistency* measures whether the images are consistent with each other, having a common topic and naturally forming a story, rather than looking like 5 independent images.
- Semantic Relevance refers to whether the images accurately reflect the captions and the characters mentioned in them.

A screenshot from our survey can be seen in Figure 7. In order to eliminate bias, for half of the examples we assign the label *Model 1* to the images generated by our model and *Model 2* to the images generated by CMOTA, while for the other half, the labels are assigned the other way around. Every pair of image-stories is annotated by two distinct annotators. We make sure not to store any personal information about the users in our survey.
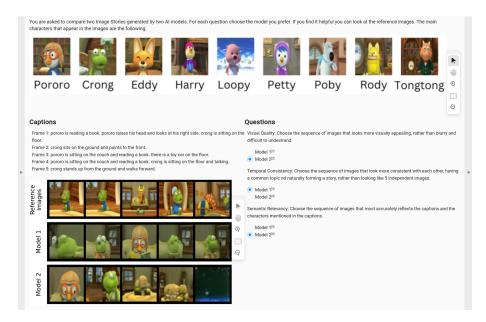
Fig. 7: A screenshot from our human evaluation survey.

G   Training Times and Resource Usage

Training We approximately spend 36 and 107 hours to train our Transformers with $d = 1024$ and $d = 2048$, respectively, on a single NVIDIA V100 (16GB). This is equivalent to (36 hours) · (16GB) = 576 (GB · hours) and (107 hours)· (16GB) = 1712 (GB · hours) of GPU usage, respectively. For reference, VP-CSV [4] reportedly uses 4 NVIDIA A100 (40GB) for 12 hours. This is equivalent to (12 hours) · (4·40 GB) = 1920 (GB · hours) of GPU usage, without taking into account that A100 is a more modern GPU than V100. CMOTA [1] does not report training resource usage.

Inference Since we performed inference for our models, as well as CMOTA, on the same GPU, we can carry out a fair comparison. Our models with $d = 1024$ and $d = 2048$ need 34 minutes and 94 minutes, respectively to perform inference for the 2208 stories of the test set. This is equivalent to 9.07 (GB · hours) and 25.07 (GB · hours). For the same task, CMOTA spent 228 minutes, which translates to 60.8 (GB · hours). It is evident that our method is significantly more time-efficient compared to CMOTA. Even the larger version of our model, is more than [2×] more efficient, during inference compared to it. This can be largely attributed to the inference scheme of MaskGIT-style transformers, that produce multiple visual tokens per step, compared to auto-regressive transformers, like CMOTA, that infer visual tokens one at a time.

H   More Qualitative Examples

Figure 8 and Figure 9 show more Story Generation examples using our *MaskGST-CG$_\pm$ /w aug. captions.* For each example we provide the images generated by our model, the input captions and the ground-truth images (original) that correspond to these captions.

Frame 1: Pororo notices the batter on the table.
Frame 2: Pororo picks up cookie cutter.
Frame 3: Pororo bakes cookies for loopy.
Frame 4: Pororo is finished with baking cookies.
Frame 5: Loopy has come back home.

Frame 1: the mean magician Eddy had to keep singing. the story ends.
Frame 2: Poby is holding a book. Poby says the story is done.
Frame 3: Pororo is telling the lesson with the example of Pororo and Harry.
Frame 4: Roby suggests to help friends.
Frame 5: Poby shook his hand saying good bye.

Frame 1: Pororo and Crong are very curious about what the Robot is. they are curiously watching it.
Frame 2: The robot asks Pororo and Crong what he can do for them.
Frame 3: Pororo and Crong are very surprised to hear the robot talking.
Frame 4: The robot asks again what can he do for Pororo and Crong.
Frame 5: Pororo decides to examine the robot with Crong.

Frame 1: Petty and Loopy are talking in Loopy house.
Frame 2: Loopy points Loopy broken chair.
Frame 3: Poby makes an excuse for Pororo. Loopy is angry.
Frame 4: Loopy explains why Petty is mad at Pororo.
Frame 5: Petty and Poby understands why Loopy is so mad at Pororo.

Frame 1: Pororo makes the snowman's legs. then Pororo makes the arm of the snowman.
Frame 2: Pororo makes arm of the snowman. then Pororo puts goggles to snowman's face.
Frame 3: Pororo puts mouth to the snowman's face. then Pororo puts buttons to the snowman's face to make the snowman's eyes.
Frame 4: Pororo finishes making his own snowman.
Frame 5: Poby is walking on the snow. Poby is waving.

Frame 1: Poby is continually in the air. Harry is now with Poby. Harry says that Harry will make Poby put down.
Frame 2: Harry and Poby are talking each other in the air. Harry tries to make Poby put down.
Frame 3: Harry and Poby are talking each other in the air. Poby has long been on the sky so Poby seems to be tired.
Frame 4: Harry and Poby are talking each other in the air. Harry tries to make Poby happy and make Poby put down. suddenly Harry jumps high and stops on top of Poby's head.
Frame 5: Harry is now on top of Poby's head. harry says that Harry will sing a song for Poby therefore Harry asks to Poby to hang in there.

Fig. 8: More Story Generation Examples using our model *MaskGST-CG*$_\pm$ */w aug. captions.*

Frame 1: Eddy Loopy and Petty don't want to listen Harry's song. they just want to eat cake.
Frame 2: Harry stands up on the balustrade. Harry tries to sing a song to her friends in spite of the friends' dissuading.
Frame 3: Harry starts to sing a song on the balustrade.
Frame 4: Pororo and his friends looks disgusted. they really don't want to listen harry's song.
Frame 5: Loopy petty and Crong also don't want to listen Harry's song.

Frame 1: Eddy has something with his right hands. that looks like a small fan. its color is light blue. Eddy looks happy.
Frame 2: the front side of the cannon is uncovered with snow. Eddy is walking by following the path which was covered with snow.
Frame 3: Eddy is walking through the path which was covered with snow. Eddy maybe thinks that Eddy will show Pororo what Eddy can do.
Frame 4: while walking Eddy sees that Pororo is coming to him. Eddy stops after watching Pororo.
Frame 5: Eddy is murmuring about something which is related to Pororo. Pororo is coming to Eddy with someone.

Frame 1: Petty would like to give warm tea to Eddy.
Frame 2: Eddy gives a cup of tea to Eddy.
Frame 3: Petty smiles. Eddy said thanks to Petty.
Frame 4: the tea was too salty for Eddy.
Frame 5: Eddy was surprised to hear that there is salt in the tea.

Frame 1: Pororo's friends are shaking their heads because they don't know what Pororo asked.
Frame 2: Harry flies to Pororo and say something.
Frame 3: Pororo ponders something for a moment and Eddy and Petty also walk to Pororo.
Frame 4: Petty and Eddy stand in a row. Eddy says something to Pororo.
Frame 5: Petty and Eddy stand in a row. Petty says something to Pororo and eddy glances at Petty and smiles.

Frame 1: now Pororo is preparing to hit Eddy. Pororo throws snowballs to Eddy. Eddy tries to avoid snowballs.
Frame 2: Eddy looks happy with his eyes wavy shaped. however Eddy is struck snowballs from Pororo.
Frame 3: Pororo seems happy with his successful hitting to Eddy. Pororo is playing snowballs with his right hands. after struck Eddy looks angry.
Frame 4: Eddy is walking toward Pororo. Pororo is looking Eddy with his curious feeling. Pororo is touching the mouth with his right hands.
Frame 5: Pororo seems depressed with his bulging mouth. Pororo is muttering with unsatisfied gesture.

Frame 1: Eddy orders the robot to stop singing and dancing. the robot stops singing and dancing.
Frame 2: Eddy is so happy and proud that Eddy made the robot.
Frame 3: Eddy is imagining how surprised all his friends would be to see the robot Eddy made. Eddy seems really excited.
Frame 4: Eddy runs out to show his robot to his friends.
Frame 5: Pororo and Crong is calling Eddy and getting into Eddy's house.

Fig. 9: More Story Generation Examples using our model *MaskGST-CG*$_\pm$ */w aug. captions.*