Vidgen: Long-Form Text-to-Video Generation with Temporal, Narrative and Visual Consistency for High Quality Story-Visualisation Tasks

Ram Selvaraj
Department of CSE
PES University
Bangalore, India
ramselvaraj01@gmail.com

Ayush Singh
Department of CSE
PES University
Bangalore, India
theayushsingh72@gmail.com

Shafiudeen Kameel
Department of CSE
PES University
Bangalore, India
skameel2002@gmail.com

Rahul Samal
Department of CSE
PES University
Bangalore, India
rsamal.b305@gmail.com

Pooja Agarwal
Department of CSE
PES University
Bangalore, India
poojaagarwal@pes.edu

***Abstract*—Generating long-form videos conditioned on large story based text input is a new and relatively unexplored task. Current text-to-video models are designed to generate short video clips conditioned on small input texts, which while temporally consistent, are severely lacking in narrative and visual consistency which are key elements of good story visualisation. This paper presents Vidgen, a pipeline that ensures Long-Form Text-to-Video Generation with Temporal, Narrative and Visual Consistency. To address the above issues Vidgen employs a new approach which leverages a Large-Language- Model (LLM) for the pre-processing of input story/script, which extracts key actions and story elements from the given script, and converts it to a standardised format which will be read and understood by the text-to-video model. The paper also proposes fixing "embeddings" (spatial-information about the appearance of the character) using a new, faster and improved textual- inversion approach with pre-trained weights, to ensure consistent looking characters. The work done in this paper will allow any traditional text-to-video model to accept large input text in the form of a story and generate high quality, temporally consistent arbitrary-length videos that have consistent looking characters for the entire duration of the video.**

***Index Terms*—Generative Artificial Intelligence, Latent Diffusion Models, Large Language Models, Text-to-Video Generation, Textual Inversion, Story Visualisation**

I. Introduction

In today's digital age, the ability to create engaging and informative video content is paramount for effective communication and storytelling. The task at hand deals with generation of lengthy videos that capture the context and content of a written story. High quality story visualisation in the form of video involves maintaining narrative, visual and temporal consistency.

Narrative consistency deals with maintaining a consistent narrative tone throughout the duration of the video, and crucially ensuring that the events of the original story are captured in their entirety in the duration of the video in order. Narrative consistency also includes the resolution of pronouns and possession words like "him", "her", "he", "they", "his" etc, along with the ability to understand long context of previous scenes to ensure that the story being portrayed is valid. These are very intricate details and require complex networks such as LLMs to fully gather the meaning and understand the context.

Visual consistency refers to maintaining a consistent visual tone throughout the duration of the video, and crucially ensuring that characters, objects and locations that appear throughout the story look the same throughout the duration of the video.

Temporal consistency refers to generating high quality videos where each subsequent frame stitches together with the previous frames in a smooth manner, thus generating realistic looking and high quality videos.

The unison of these 3 key consistency elements leads to the creation of high quality, visually appealing and compelling story based videos, and crucially allow for the task of story-visualisation through long-form video as shown in Fig. 1.

II. Related Work

Visual consistency refers to the consistent appearance of characters, backgrounds etc. that appear throughout the story. This task has been achieved in traditional story-visualisation tasks, such as [1] and [2]. These approaches tackle the task of story-visualisation by the generation of still images that represent key scenes in an input story. In other words, they can create a slideshow of images that depict the story. They achieve consistent visuals due to their limited training and testing data, where they have prior knowledge on what characters will appear before the input script is even received. In other

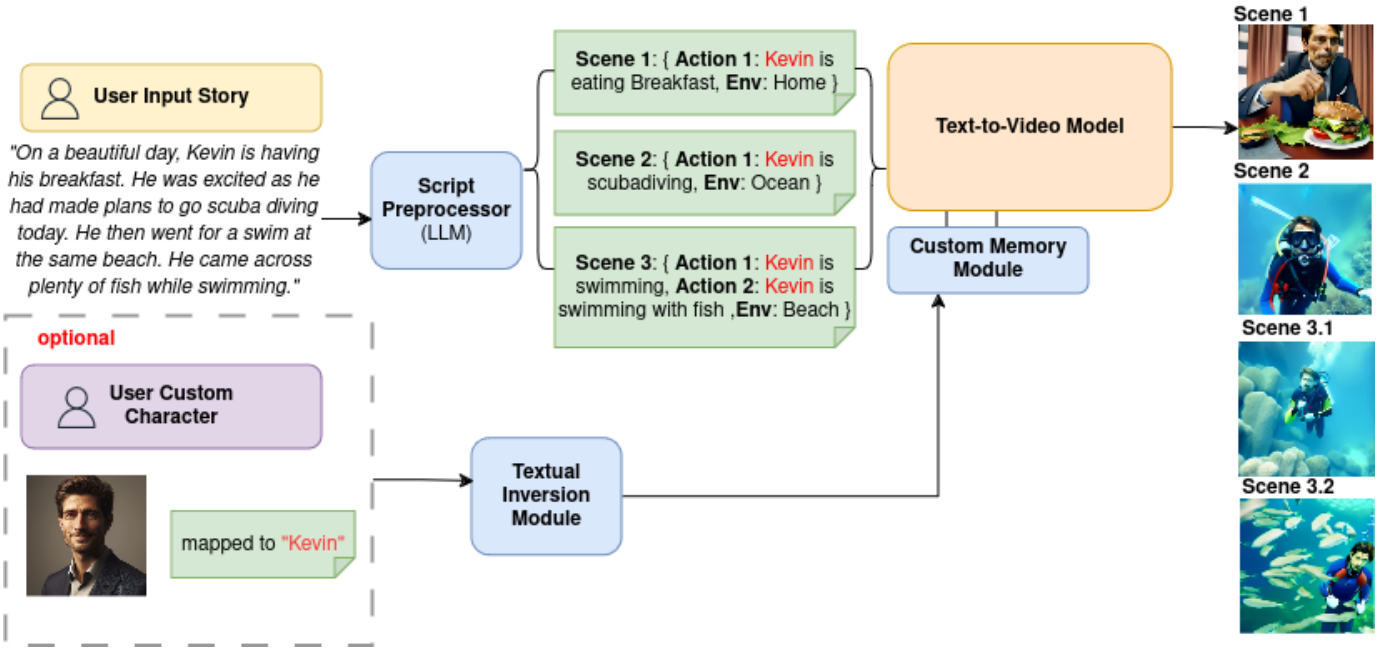


Fig. 1: An overview of the Vidgen Pipeline.

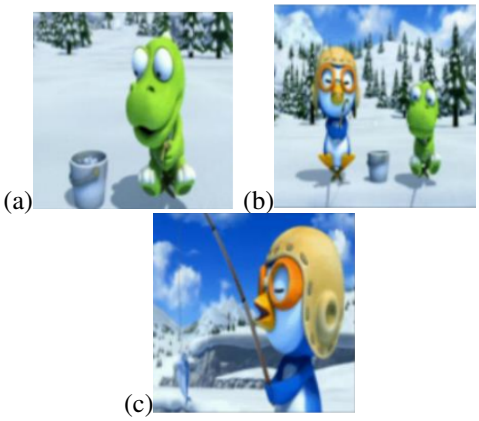


Fig. 2: (a) *"**Crong** is looking at the bucket"* (b) *"**Pororo** and **Crong** are fishing together"* (c) *"**Pororo** has a fish on his fishing rod"*
StoryGAN [1] output, example of Visual Consistency

Fig. 3: (a) *"**John** eating a burger"* (b) *"**John** holding a glass of milk"*
John is generated as two completely different individuals, example of Visual Inconsistency

words, they are limiting their generation capabilities only to characters that they can generate consistently. An example of StoryGAN's visually consistent generation is showcased in Fig. 2

Existing text-to-video models [3, 4, 5, 6] and techniques face significant limitations in generating long-form videos with consistent visual and narrative flow. The current offerings are very good at generating high quality temporally consistent realistic looking clips of short duration, that are strictly conditioned on short text-inputs. One can individually take sentences from the original story and have an existing text-to-video model generate videos for each and append them to each other. There are several drawbacks with this approach however, **i)** the model cannot resolve pronoun and possession ambiguities like "he","she", "they", "his", "hers" etc. and **ii)** suppose the model sees the same character in two different scenes, it will generate two different looking individuals across these two scenes. As shown in Fig. 3, the character "*John*" appears as two different looking individuals across the two scenes. This visual inconsistency occurs because when the model's tokenizer reads the token "*John*", it will point to a general learned embedding, not an embedding that explicitly contains information of the specific appearance of "*John*". The general learned embedding would only contain high level information like 'male', 'human' etc, and not more detailed information like 'blonde','blue eyes', 'white shirt' etc. Therefore, while the generated videos are temporally consistent, the lack of the other 2 key consistency factors makes existing text-to-video models not viable for the proposed task.

The task of long-form video generation has been extensively

worked on [7, 8, 9, 10], although not in the sub-domain task of text-to-video. Researchers [9] have been able to develop powerful networks with the ability to generate arbitrary length long videos, however none of these networks have the provision to follow any kind of text-guidance, let alone handle the nuances and complexities of long input scripts like stories.

III. Proposed Methodology

The approach taken with Vidgen involves using an existing temporally consistent, high fidelity text-to-video model, and modifying it with novel additions to support the generation of i) visually consistent and ii) narratively consistent videos conditioned on large story based input text.

Attempting to fix the issue of Visual Consistency, and taking inspiration from [1] and [2], both approaches that only support a fixed set of characters present in their training data, in other words, already know what the user wants to create. Therefore, taking an approach that involves identifying what characters are present in a user's input story beforehand, Vidgen can ensure that those characters always get generated in a visually consistent manner. In clearer terms, Vidgen proposes identifying the characters that are present in the user's story right as they upload their story (before the video generation process begins). The approach taken to ensure this consistency involved fixing the embeddings of the character that will be referred to during the video generation process, with embeddings that contain all the relevant information that ensures the character appears consistent (character's features, clothing details etc) regardless of where the character is being mentioned in the story.

To solve the issue of narrative consistency and long-form arbitrary length video generation, Vidgen employs a Large Language Model (LLM). An LLM with its ability to read large sized input scripts, understand it through their powerful learned word embeddings, is ideal for this task. The approach here involves having the LLM generate a standardised output which contains the key actions and scenes that have been extracted from the input story, in order. This enables Vidgen to leverage existing text-to-video models along with their high-fidelity generations to generate these actions and scenes. These individually generated scenes can then be stitched, thus creating arbitrary length videos that contains the entire story. This approach gives the best of both worlds, by taking advantage of the high-fidelity, highly relevant (matches accurately with guiding text) outputs and remove their short-form limitations by generating arbitrary length videos.

A. *Modules/ Implementation Details*

Vidgen is an end-to-end pipeline that consists of several modules, all of which work in harmony to achieve the task of long-form story visualisation. The architecture of Vidgen is showcased in Fig. 4. The paper will discuss each of the modules in the following subsections.

1) ***Script Preprocessor****:* Vidgen's Script Preprocessor Module leverages a Large Language Model (LLM). An LLM was chosen for its ability to thoroughly analyse input scripts ,given that they have a network large enough to pay attention to a large input and understand them through their powerful learned embeds. Actions, characters, and situations are among the essential story aspects that the LLM is excellent at extracting. This paper is employing Meta's Llama-2-7B-chat model [11], an instruction following, 7 billion parameter, open source, locally inferable model. Llama 2 is a collection of pretrained and fine-tuned generative text models ranging in scale from 7 billion to 70 billion parameters. The chosen Llama-2-7B-chat is further fine tuned to receive an input script, convert it into a standardised format, and additionally output a list of all the characters that appear in the story. Essentially, the fine-tuned LLM acts as a 'Director' or a 'Screenplay Writer', that gives prompts that Vidgen's text-to-video generator can understand. The standardised format (showcased in Fig. 5) the model is using to extract and contain the information of the actions and scenes in the input script is defined as follows- The LLM has been fine-tuned and instructed to read the input story, and generate a JSON output that contains the story broken up into a series of **scenes**, divided based on the environment in which they occur, while also maintaining the order of events in the original story. The LLM is also tasked with reading the story and returning a list of characters that appear in the story, which is used in the following modules to ensure visual consistency of characters.

2) ***Fast Textual Inversion Module****:* Textual Inversion is a technique developed by [12], that enables identifying what single embedding represents an entire concept, learnt backwards from the images of the concept. For example, if an image of a specific cat is available, what embeddings represent that specific cat, which includes all its descriptive features, in the text-to-image model's embeddings space. This process of identifying the right embeddings that represent the concept can be extended to even learn the style of images, not just the look of objects, and chiefly for Vidgen's use-case, the look of a character.

Vidgen uses Textual Inversion to learn the descriptive features of the characters present in the story, from images of the characters, and compress these features into a single custom learned embedding, which can be mapped to the respective character's token. In doing so, Vidgen is ensuring that anytime the required character has to be generated as part of their respective scenes, which will be identified by the presence of that character's token (usually their name), the token will map to the custom embedding that contains all the required information about the descriptive features of the character, thus ensuring visually consistent looking characters in any scene, throughout the video.

Textual Inversion involves using a Latent Diffusion Model [13], and its respective tokenizer [14] to learn inn a backwards fashion what embeddings ($c_\theta(y)$, refer Eq. 1) represent the image. It does this by first guessing a random embedding and generating an image. The loss function (Eq. 1) between the two images can be optimized (Eq. 2) to find the exact embedding ($c_\theta(y)$) which can generate an image that closely matches the

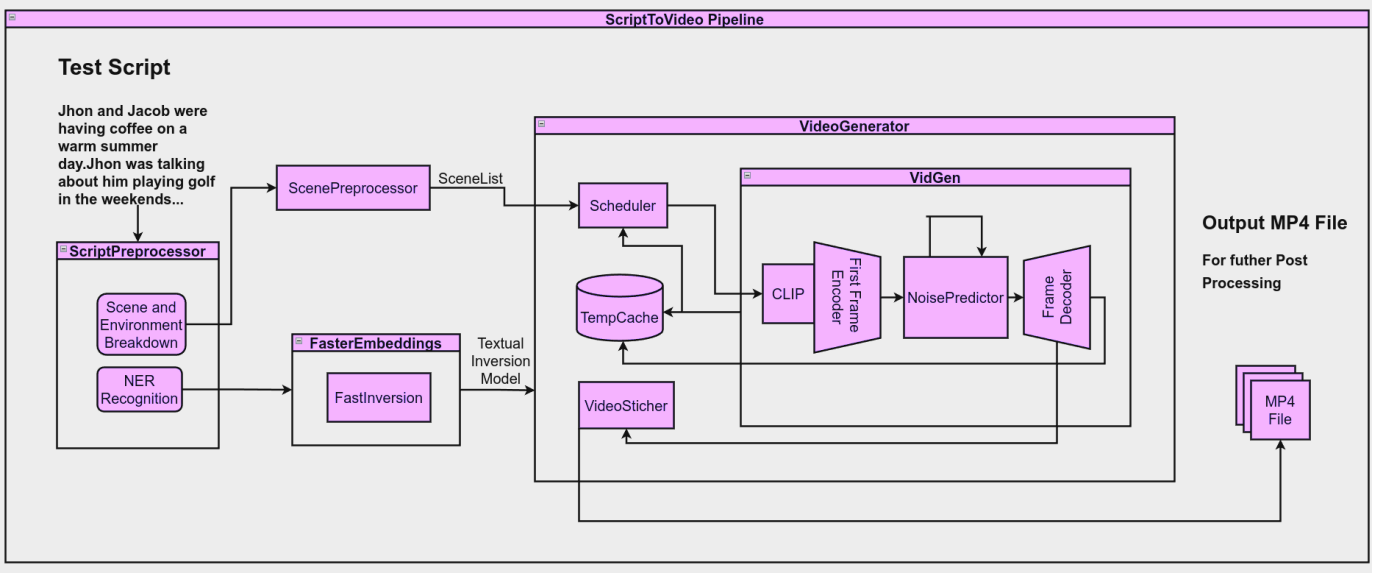


Fig. 4: *Vidgen Architecture*

```
{       "Scene 1": { [
                "Action 1": "Jack is taking a walk",
                "Action 2": "Emily was watering her garden"
                .
                .
                .
                "Action n": "Jack and Emily look at each other and smile"
                 ],
                "Env": "City"}]
        "Scene 2": {[
                "Action 1": "Jack and Emily are walking together"
                .
                .
                .],
                "Env": "Garden"}]
        .
        .
        "Scene N": {[...
        .
}
```

Fig. 5: *Output of LLM, fine-tuned to match the standard format*

original image.

$$L_{LDM} := \mathbb{E}_{z\sim\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta\left(z_t, t, c_\theta(y)\right)\|_2^2\right], \tag{1}$$

$$v_* = \arg\min_v \mathbb{E}_{z\sim\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta\left(z_t, t, c_\theta(y)\right)\|_2^2\right] \tag{2}$$

The authors of [12] assume random embeddings in the beginning and work from there trying to identify the right embeddings. However, this paper has found an optimization where in using a checkpointed weight in the latent diffusion model that is used to learn the embedding, can significantly speed up this process. The checkpointed weight will start from an embedding that closely resembles a human face, so trying to find the right embedding is much faster as the corrections made are far smaller. This approach reduces runtime from 1hr per character to 5 mins per character. This technique has been named as FastInversion, visible in Fig. 4

This paper is incorporating Textual-Inversion for two aspects i) Learning a few characters embeddings and storing it, i.e general purpose characters, such as attractive models and paid actors that people will be willing to select and use in their videos, and ii) learning new custom characters that users can submit at the time of uploading the script, thus giving them the added customization to have their custom characters appear in the video. This module is later combined with Vidgen's Custom Memory Module (Sec. III-A4).

*3) **Video Generator**:* Extensive searching in the domain of text-to-video lead to the discovery of the paper [5], discussing an open-source model that can generate high quality, albeit short video clips, but most importantly, videos that were highly conditioned on the text input. The approach taken by the authors of this paper involves decomposing the diffusion process. The fundamental philosophy relies on the fact that a video is a sequence and frames, and for shorter instances, all frames are largely just a variation of the first frame. Exploiting this fundamental aspect, the authors have devised a technique where they generate the first frame using a typical text-to-image model. With the first frame now generated, they begin generating the remaining frames for the sequence. Here is where the author's novelty of decomposed diffusion kicks in. Given that this is a diffusion model, every diffusion model during inference will denoise from pure white noise into the final image, so at a time step closer to the final time step (for example t=40 out of 50 total steps), the image is mostly denoised bar some extra noise. The authors call this the **base noise** (x_0). They then compute the **residual noise** (x_i), i.e the remaining noise as a function of this base noise with some learned coefficients. The learned coefficients are conditioned to follow the text-prompt guidance as they are trained on video-text action pair datasets, therefore giving the model the ability

to generate videos that follow the actions specified in the text prompt, while still staying temporally consistent, and visually consistent only for that short scene. Keep in mind this visual consistency does not extend across different prompts. In other words while this model has **'intra scene'** consistency, there is no **'inter scene'** consistency. Therefore, with this model's architecture chosen as the base video generator for Vidgen, given its open-source nature, combined with Vidgen's Fast Inversion Module (refer Sec. III-A2) and Custom Memory Module (refer Sec. III-A4) leads to the generation of **'inter scene'** visually consistent videos that are also temporally consistent and of a high quality.

*4) **Custom Memory Module**:* This module is responsible for handling the token and embedding mapping for each , character, appropriately editing, re-adding, and deleting the correct tokens and embedding mappings to Vidgen's tokenizer. It identifies the tokens in a given scene prompt and dynamically loads only the corresponding embeddings at runtime. This was done as an optimization step to further reduce memory consumption during runtime. This module is combined with the previously discussed Fast Textual Inversion Module (refer Sec. III-A2), and this combined module will be referred to as 'Custom Textual Inversion Memory Module' for the remainder of this paper.

IV. Experiments/Results

This section discusses the results of the different evaluations run on Vidgen, for the task of long form story visualisation with temporal, narrative and visual consistency.

A. *Qualitative Results*

All applications of Vidgen, show that it successfully generates videos that are high quality with near realistic visuals, and that are temporally consistent with smooth motions of the characters.

In the task of story visualisation through text-to-video, Vidgen excels in maintaining consistent looking characters, backgrounds and narratively consistent videos where the events occur in the same order as in the original input story, and the correct characters performing the right actions.

*1) **Visual Consistency**:* Maintaining visual consistency across different scenes is one of Vidgen's key novel features. Vidgen can successfully generate different video sequences with characters that look the same across the different sequences. For instance, attempting to solve the problem showcased earlier in Fig. 3, where the character *"John"* appeared as two different looking individuals across the two different input prompts, Vidgen can successfully generate clips from the two different input prompts, while maintaining a consistent appearance for *"John"*. Refer to Fig. 6

*2) **Custom Character**:* Vidgen can incorporate custom character designs, using reference images uploaded by the user. For instance, if the user uploads a reference image (Fig. 7(a)) for the custom character *"Hailey"*, using Textual Inversion, Vidgen will learn how to represent *"Hailey"* in it's embedding space and assign the learned embedding to the token *"Hailey"*. This results in successful generation of video clips with the character looking consistent with the reference image, while now performing all the actions mentioned in the input story. Refer to Fig. 7

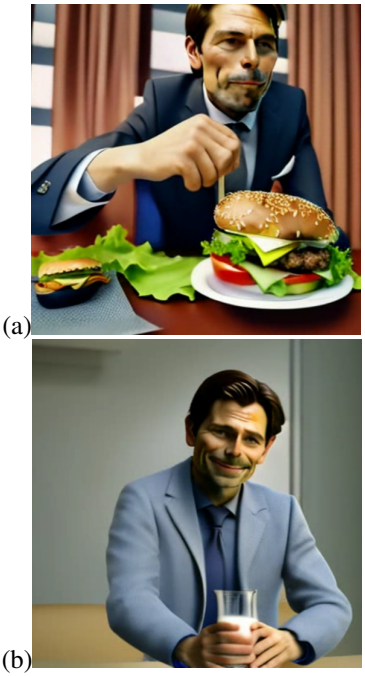
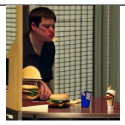
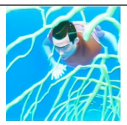
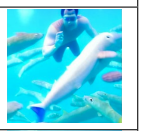
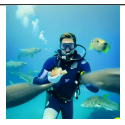
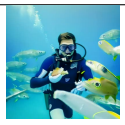
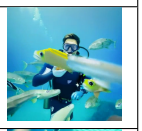
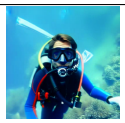
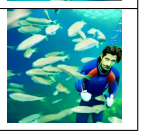


Fig. 6: (a) *"**John** eating a burger"* (b) *"**John** holding a glass of milk"*

Fig. 7: (a) Reference image for **Hailey** (b) *"**Hailey** eating a burger"* (c) *"**Hailey** swimming underwater"*

B. *Ablation Study*

An ablation study was performed on the two main novel aspects of Vidgen, i) custom textual inversion memory module for maintaining visual consistency and ii) script preprocessor for maintaining narrative consistency. Both modules are crucial for Vidgen's success in the task of story visualisation. The

Vidgen Scriptprocessor	Vidgen Custom Textual Inversion Memory Module	**Scene 1**	**Scene 2**	**Scene 3.1**	**Scene 3.2**
✗	✗				
✓	✗				
✗	✓				
✓	✓				

This table illustrates an ablation study conducted on the Vidgen system, exploring the effects of different configurations involving the Vidgen Scriptprocessor and the Vidgen Custom Textual Inversion Memory Module across four scenes. The checkmarks (✓) and crosses (✗) denote the presence or absence of these modules. The visual representations in each cell depict the output generated by the Vidgen system based on the specified module configurations for the respective scenes. The original input story provided to Vidgen is as follows- *"On a beautiful day, Kevin is having his breakfast. He was excited as he had made plans to go scuba diving today. He then went for a swim at the same beach. He came across plenty of fish while swimming."*

TABLE I: **Ablation Study on Vidgen**

qualitative results are present in Table I. The table is laid out with the first two columns indicating the presence or absence of this paper's novel additions to Vidgen's base video generator. The test was performed with an input story that consisted of 4 events. The following is a qualitative analysis of the results of the ablation study. In the first row, without Vidgen's script Preprocessor or custom textual inversion memory module, the base video generator (same as any generic text-to-video model) alone is unable to differentiate the separate events that occur in the story and instead creates a sequence that tries to capture all the information in the story in almost a concatenated fashion. It completely fails to capture the characters in the story, let alone attempt to maintain any visual consistency. For the next test, the base video generator along with Vidgen's Script Preprocessor (second row), the results are already far better. Already the 4 different events occurring in the input story have been understood and visualised. The events also occur in the same order as the input story, ensuring narrative consistency. The character *"Kevin"* has also been recognised and appearing in all the scenes, however clearly the appearance of *"Kevin"* changes in every scene that he appears in. For the third test, the base video generator Vidgen's Custom Textual Inversion Memory Module, but doesn't have access to Vidgen's Script Preprocessor (third row). The result is a video that has the character *"Kevin"* and he has a consistent appearance throughout, however the video is once again in concatenated fashion that has all the content of the story compressed together, where actions performed by the character in separate scenes in the original story are all appearing together, as the Script Preprocessor isn't available. In other words, visual consistency is present but there is no narrative consistency. Finally for the fourth test, the base video generator is given access to Vidgen's Custom Textual Inversion Memory Module and Script Preprocessor (fourth row). Clearly from the results, it is visible that the character *"Kevin"* has been recognised, his appearance remains constant, as well as the story being played out as separate scenes as originally present in the input story. Therefore, it can be concluded that Vidgen along with its novel additions has successfully visualised the input story.

C. *Quantitative Results*

In Fig. 8 the graph illustrates the effects of Vidgen's Custom Textual Inversion Module and the Script Preprocessor with respect to character consistency of the given model. The x-axis represents the frames generated over time intervals, while the y-axis quantifies the similarity scores of a given character between different scenes, with his appearance in the first scene. The graph represents the consistency in the appearance of the characters throughout the duration of the video, concurrently comparing Vidgen to a basic text-to-video generation process. To calculate the similarity scores, a facial recognition system called DeepFace given by [15] has been used. A threshold score of 0.65 for two characters to be similar which is shown by the dotted line. This type of plot effectively conveys the fluctuations and character inconsistencies over time. The red-line represents no optimisations, here when the scene changes the similarity scores instantly drops below the threshold, and stays there for the remainder of the video. The effects of the Custom Inversion Module are shown by the orange line, which stays above the threshold for the majority of the frames, but has dips when the non character objects mentioned in the script overpopulate the given frame. The

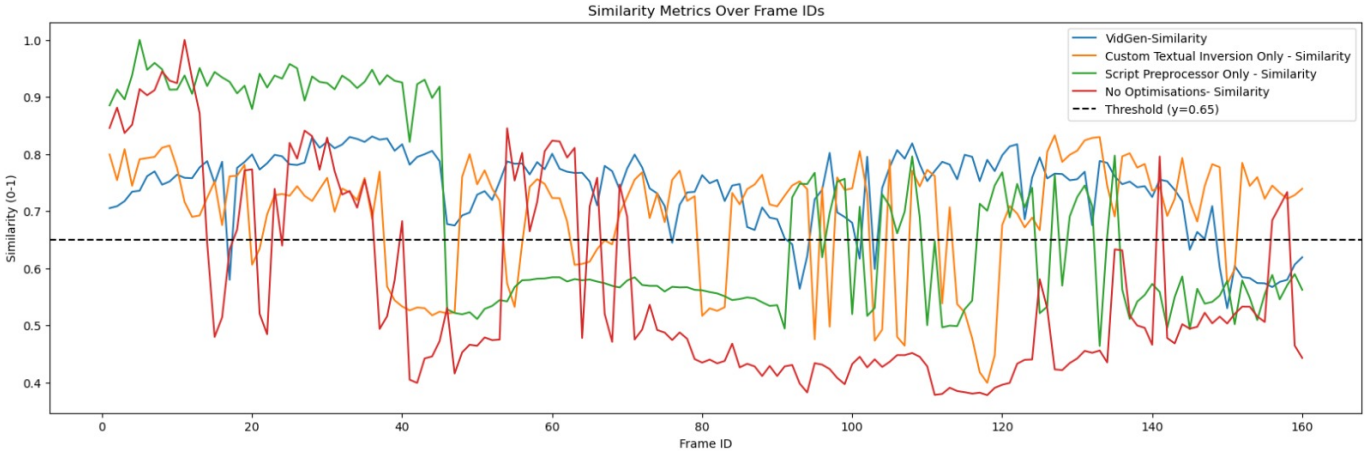


Fig. 8: *Quantitative comparison of Vidgen*

Script Preprocessor makes sure that one given scene has one consistent character throughout (visual consistency within a scene), which is explained by the flat segments in the red-line, but is still below the threshold as there is no visual consistency. Vidgen along with its novel additions is represented with the blue line and stays well above the threshold for the entire duration of the video, therefore showcasing that the Script Preprocessor and Custom Textual Inversion Memory Module working in tandem creates videos with visually consistent characters.

V. Conclusion

The idea behind Vidgen's creation is to close the gap that exists between text-based storytelling and video-based story visualisation. Vidgen can handle large, complex scripts and makes the task of producing long-form, high-quality story-based videos possible. The architecture of Vidgen combines several state-of-the-art modules, each designed to provide smooth integration and synchronisation in the pipeline for creating videos. Together, the Script Preprocessor, Custom Textual Inversion Memory Module, and the Video Generator modules work in tandem as Vidgen's backbone. The key novelty with Vidgen lies in i) its script processing which leverages an LLM and takes away the ambiguity and complexities, the resolution of which would have to be otherwise learnt by extensive training of text-to-video models on currently non-existent datasets of story based text-videos, and ii) crucially its ability to maintain consistent looking characters throughout the entire duration of the video. Vidgen and it's design approach are both adaptable, modular and can be repurposed across various fields of Generative AI, and new downstream tasks. Although it is vital to be aware of the possible risks with respect to video generation, Vidgen offers unquestionable advantages that make it a valuable tool for improving various fields like education and entertainment and a variety of user groups like individuals, companies, and organisations.

References

[1] Y. Li, Z. Gan, Y. Shen, J. Liu, Y. Cheng, Y. Wu, L. Carin, D. Carlson, and J. Gao, "Storygan: A sequential conditional gan for story visualization," 2019.

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

[3] U. Singer, A. Polyak, T. Hayes, X. Yin, J. An, S. Zhang, Q. Hu, H. Yang, O. Ashual, O. Gafni *et al.*, "Make-a-video: Text-to-video generation without text-video data," *arXiv preprint arXiv:2209.14792*, 2022.

[4] W. Hong, M. Ding, W. Zheng, X. Liu, and J. Tang, "Cogvideo: Large-scale pretraining for text-to-video generation via transformers," 2022.

[5] Z. Luo, D. Chen, Y. Zhang, Y. Huang, L. Wang, Y. Shen, D. Zhao, J. Zhou, and T. Tan, "Videofusion: Decomposed diffusion models for high-quality video generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 209–10 218.

[6] Y. Li, M. Min, D. Shen, D. Carlson, and L. Carin, "Video generation from text," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[7] S. Ge, T. Hayes, H. Yang, X. Yin, G. Pang, D. Jacobs, J.-B. Huang, and D. Parikh, "Long video generation with time-agnostic vqgan and time-sensitive transformer," in *European Conference on Computer Vision*. Springer, 2022, pp. 102–118.

[8] T. Brooks, J. Hellsten, M. Aittala, T.-C. Wang, T. Aila, J. Lehtinen, M.-Y. Liu, A. Efros, and T. Karras, "Generating long videos of dynamic scenes," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 769–31 781, 2022.

[9] Y. He, T. Yang, Y. Zhang, Y. Shan, and Q. Chen, "Latent video diffusion models for high-fidelity long video generation," *arXiv preprint arXiv:2211.13221*, 2023.

[10] F.-Y. Wang, W. Chen, G. Song, H.-J. Ye, Y. Liu,

and H. Li, "Gen-l-video: Multi-text to long video generation via temporal co-denoising," *arXiv preprint arXiv:2305.18264*, 2023.

[11] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[12] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or, "An image is worth one word: Personalizing text-to-image generation using textual inversion," *arXiv preprint arXiv:2208.01618*, 2022.

[13] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[15] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.