Autonomous Multi-Modal Story Generation with BERT, GPT-3.5-turbo-0125, and Stable Diffusion

Mohtashim Butt
LUMS
School of Science and Engieering
Dept. of Computer Science
24100238@lums.edu.pk

Humaira Fasih Ahmed
LUMS
School of Science and Engieering
Dept. of Computer Science
24100077@lums.edu.pk

Rehan Ahmed
LUMS
School of Science and Engieering
Dept. of Physics
24100219@lums.edu.pk

Rija Sajjad
LUMS
School of Science and Engieering
Dept. of Electrical Engineering
24100166@lums.edu.pk

Taha Sarwat
LUMS
School of Science and Engieering
Dept. of Computer Science
25100287@lums.edu.pk

Jahanzaib Khursheed
LUMS
School of Science and Engieering
Dept. of Computer Science
24100257@lums.edu.pk

***Abstract*—This paper presents our novel approach to a multi-modal conversational story generation model, wherein the user provides initial context for story, and initial character, event, and place. Our system then generates coherent and long, multi-paragraph stories on its own thereafter. We leverage fine-tuned GPT-3.5-turbo-0125 for storyline prediction, raw GPT-3.5-turbo-0125 for paragraph generation, and Stable Diffusion (DALL-E-3) for visually explaining the story. Our approach dynamically adjusts story elements based on evolving contexts, predicting a storyline based on multiple-choice question-answering problems.**

GitHub: https://github.com/MohtashimButt/conversational-story-generation

I. Introduction

Storytelling stands as an ancient tradition throughout human history. With the advancement of AI technologies, generative models have emerged as significant players in this realm, contributing to the development of various methods for story generation. Automated story generation involves using an intelligent system to produce fictional stories with minimal human involvement. While SOTA LLMs such as GPT-2 [1] and BART [2] have showcased impressive capabilities in generating text, the art of storytelling requires more than merely stringing together coherent sentences; it demands an understanding of plot development, character dynamics, and thematic coherence – elements that are often overlooked in conventional text generation tasks. We have observed that conventional sequence-to-sequence (seq2seq) models, as described by [3], when utilized for hierarchical story generation, tend to devolve into language models that exhibit minimal responsiveness to the provided writing prompt. The motivation for our project arises from these shortcomings.

Conversational story generation represents a paradigm shift in narrative construction. Unlike traditional story generation approaches that rely on predetermined plot structures or templates, conversational story generation fosters dynamic interactions between user requirements and AI agents, allowing for personalized and contextually relevant narratives.

Our work is primarily influenced by the foundational work presented in [4]. We propose a methodology wherein our system autonomously plans and generates stories given the initial plot and list of characters involved. Our approach entails continuously predicting storyline elements via fine-tuned GPT-3.5-turbo-0125 [6], generating paragraphs via raw GPT-3.5-turbo-0125 [6], and generating the corresponding image via Stable Diffusion [7] in real-time without human intervention. Our system maintains coherence throughout the narrative, ensuring engagement from start to finish.

The fine-tuned version of GPT-3.5-turbo-0125 is just used for predicting storyline entities—such as characters, events, and places.

II. Methodology

This section discusses the approaches we used to generate stories coherently. We utilized three types of models:

- Fine-tuned GPT-3.5-turbo-0125 (for story guidance).
- Raw GPT-3.5-turbo-0125 (for paragraph generation)
- Diffusion Model (DALL-E-3 [8])

We used Storium Dataset [9] and did its extensive preprocessing. Storium contains 6K lengthy stories (125M tokens) with fine-grained natural language annotations (e.g., character goals and attributes) interspersed throughout each narrative, forming a robust source for guiding models.

A. Dataset

The Storium dataset consists of 5743 stories, including a large corpus of 25,092 scenes, containing a diverse range of stories comprising a wide range of narrative scenarios, characters, and plotlines. Within the Storium dataset, each story is structured into scenes, each representing a discrete narrative unit. It contains cards that specify each character's

role, and then these characters have their own background stories and other contextual attributes.

B. Preprocessing

Our preprocessing pipeline is designed to structure and organize input data effectively for story generation. Essential components include data structures like Trim, IndexedSet, and IndexedDict, which facilitate data trimming, uniqueness enforcement, and efficient indexing. These structures contain our scene information and characters and help us form a flow of the story (which is difficult otherwise). Using these structures and some user-defined data classes, we find the scenes associated with each character, the location of the scenes, and the event in the scene. These are

Our preprocessing pipeline also involves the implementation of a SpecialToken class, crucial for identifying and demarcating different parts of the narrative, such as characters and actions, using special tokens. This enables meaningful segmentation of the data. The encode function performs standard tokenization, while the encode special function extends this functionality to handle special tokens and create segments. Furthermore, our preprocessing pipeline includes the use of data classes such as ProcessedStory, EntryInfo, and CharacterInfo, which organize and structure the processed data into coherent formats suitable for downstream processing and model training. This ensures that the input data is properly structured, segmented, and formatted to facilitate effective utilization by our story generation system. In our case, fine-tuning involves exposing BERT to a corpus of character-driven stories from the Storium dataset, allowing it to learn the relationships between characters, events, and plot developments. Through this iterative process, BERT gradually refines its representations to better align with the narrative structures present in our dataset.

C. Finetuning

The fine-tuning process for GPT-3.5-turbo-0125 involves utilizing a Multiple Choice Question Answer (MCQA) approach. This methodology entails providing the model with a contextual paragraph and training it to predict each en-tity—character, event, and place—contained within it. Each entity's prediction is trained individually. For every entity, three answers are provided for a corresponding question, two of which are incorrect. Fig-1 illustrates the MCQA approachin the fine-tuning. By iteratively adjusting its parameters basedon the patterns present in our dataset, GPT-3.5-turbo-0125 gradually refines its representations to capture the essence of character-driven narratives.

This fine-tuning enables our model to incorporate contextual information from input characters, thereby enhancing its ability to generate coherent and contextually relevant storylines.

D. Guidance Model: Fine-Tuned GPT-3.5-turbo-0125

We get the place, character, event, and corresponding context (we'll refer to these as "entities") from the fine-tuned GPT-3.5-turbo-0125 which are then fed to the raw GPT-3.5-turbo-0125 to generate the subsequent paragraph.

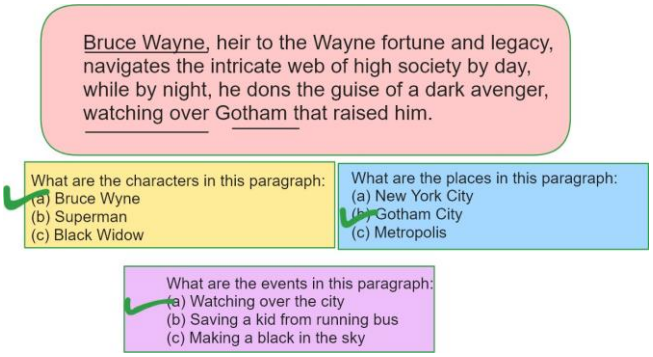
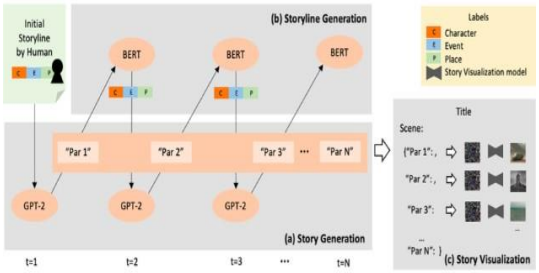


Fig. 1: MCQA Approach in GPT-3.5-turbo-0125 Fine-tuning

E. Generation Model: GPT-3.5-turbo-0125

In our generation model, we leverage the GPT-3.5-turbo-0125 model from OpenAI to generate the subsequent paragraph based on the character, place, event, and context it gets from fine-tined version GPT-3.5-turbo-0125. That generated paragraph from the raw GPT-3.5-turbo-0125 is then fed to the fine-tuned GPT-3.5-turbo-0125 (for the next paragraph's entity prediction). Raw GTP-3.5's generated paragraph is also fed to the Stable Diffusion Model (for image generation). This cycle continues until we get the concluding entities from the fine-tuned GPT-3.5-turbo-0125. The reason for using two GPT-3.5-turbo-0125 models (one fine-tuned and the other raw) is to maintain coherence in the long story with one acting as a guidance model (fine-tunde GPT-3.5-turbo-0125) and the other one acting as paragraph generator (raw GPT-3.5-turbo-0125).

1) Diffusion Model: DALL-E-3: To further enrich the reader's immersion and deepen their engagement with the narrative, our system incorporates visual elements in the form of images corresponding to each paragraph's theme. For this, we employ a diffusion-based model, which operates by diffusing contextual cues extracted from the text to produce visual representations that resonate authentically with the storyline. Additionally, we utilize CLIP, a powerful vision-language model that will generate contextual embeddings, to guide the diffusion process.



III. Deployment

We used Python Flask with JavaScript framework to make a single-page application for our end-to-end system. The GPT-3.5-turbo-0125 API integration, along with the Stable Diffusion Model (DALL-E-3) integration, was challenging, for

which we bought $20 credits from OpenAI to get them running on their cloud instead of our own local servers. We deployed the front end of our application on Vercel and the back end in Pythonanywhere.
Deployed System: ADD DEPLOYED LINK HERE

IV. Experiments & Iterations

A. Shift from BERT to GPT-3.5-turbo-0125

Initially, we opted for BERT fine-tuning to utilize it as a story guidance model following the MCQA approach. However, we encountered several challenges. Primarily, BERT fine-tuning proved inefficient, demanding substantial time and computational resources, especially when processing a dataset comprising lengthy stories totaling 6,000 examples with 125 million tokens. Moreover, BERT can handle the token length limit of only 512 tokens, but the average context description in the Storium dataset went over 4096, which BERT couldn't have handled. We had to truncate or summarize the tokens, which would have led to some information loss.

Considering the above shortcomings, we went with GPT-3.5-turbo-0125 instead for fine-tuning. Unlike BERT, GPT-3.5-turbo-0125 does not require task-specific fine-tuning; instead, it generates human-like text based on the input prompt provided. Its strength lies in its decoder-only architecture, which produces coherent and contextually relevant text across various tasks, including text completion, translation, and question-answering, without requiring task-specific training data.

B. Shift from stabilityai/sdxl-turbo to DALL-E-3

`stabilityai/sdxl-turbo` is a diffusion model that cannot render legible text [10]. Moreover, the autoencoding part of the model is lossy, and the faces and people in general may not be generated properly. Furthermore, sdxdl needs GPU to process so it required a lot of computational power from the CPU-only machines that would've hindered the deployment process. To counter all of these shortcomings, we iterated our decision to use a stable diffusion model and opted for DALL-E-3 (since we had already bought credits for OpenAI API).

V. Reflections

The collaborative utilization of the raw GPT-3.5-turbo-0125 alongside the fine-tuned 3.5-turbo-0125 model yielded impressive results. Furthermore, our DALL-E-3 diffusion model generated satisfactory images, albeit with the potential for increased accuracy and realism, given fewer cost constraints. Regrettably, resource limitations precluded the adoption of state-of-the-art text-to-image models with superior performance compared to DALL-E-3.

In the development process, we used OpenAI's API for fine-tuning. This approach offered several advantages that significantly enhanced the effectiveness of our narrative generation process. Since the model was pre-trained on generic text and not specifically on the Storium dataset, we opted to fine-tune it using our own data. This approach provided a viable alternative to using BERT, which presented challenges due to its extensive training time and token length constraints. BERT's maximum token limit of 512 proved insufficient for our project, particularly in capturing the initial scene context, which often exceeded 1000 tokens. Despite attempts to summarize the context, BERT struggled to handle the volume of tokens effectively. However, transitioning to GPT-3.5-turbo-0125, with its extended token limit of 16385, proved instrumental in mitigating these limitations. The increased token capacity enabled GPT-3.5-turbo-0125 to understand scene contexts and generate outputs with enhanced coherence more comprehensively.

Additionally, the requirement for significant computational resources, particularly GPU capacity, was a notable constraint in the context of utilizing BERT for our narrative generation pipeline. Given the limited availability of GPU resources and the high computational cost associated with BERT's tokenization, integrating BERT was impractical. As a result, we opted for OpenAI's platform, which offered a more accessible and scalable solution that avoided the need for extensive GPU infrastructure.

Furthermore, the decision to fine-tune OpenAI's model offered several advantages over using BERT as the storyline guidance model. By passing scene-related questions to OpenAI during the fine-tuning phase, we ensured that the model learned to predict character entities, locations, and other scene attributes directly from our data. This approach bypassed the need for intermediary summarization using BERT, which often resulted in a loss of context. Instead, fine-tuning OpenAI directly on our dataset allowed for a more nuanced understanding of scene contexts and facilitated the generation of more coherent outputs. The flexibility and adaptability of OpenAI's platform, coupled with its ability to handle larger token inputs, contributed significantly to the success of our project. Overall, the decision to fine-tune OpenAI's model emerged as a key factor in overcoming challenges related to token length constraints and optimizing the narrative generation process.

We could have altered the hyperparameters like batch sizes and learning rates in the training process to improve the models' convergence and optimized our preprocessing pipeline such as we could refine our model for extracting scenes and characters from the raw text data by using POS tagging to extract scene boundaries and character mentions within the text. We could use character based tokenization where out of vocabulary words will be handled more efficiently where rare words can be represented using their constituent characters. To overcome these tokenization constraints, we could experiment with segmented input strategies. We could segment the input text into smaller chunks based on narrative structure, like chapters, which could help us with issues like token length limitations while preserving contextual coherence. We can also use T5, a text-to-text transfer transformer or XLNet, for narrative generation since their ability to find dependencies between tokens can give us more coherence.

A. Demonstration and Evaluation

The front-end (UI) of our system looks something like shown in Fig: Once you press the generate button, it generates

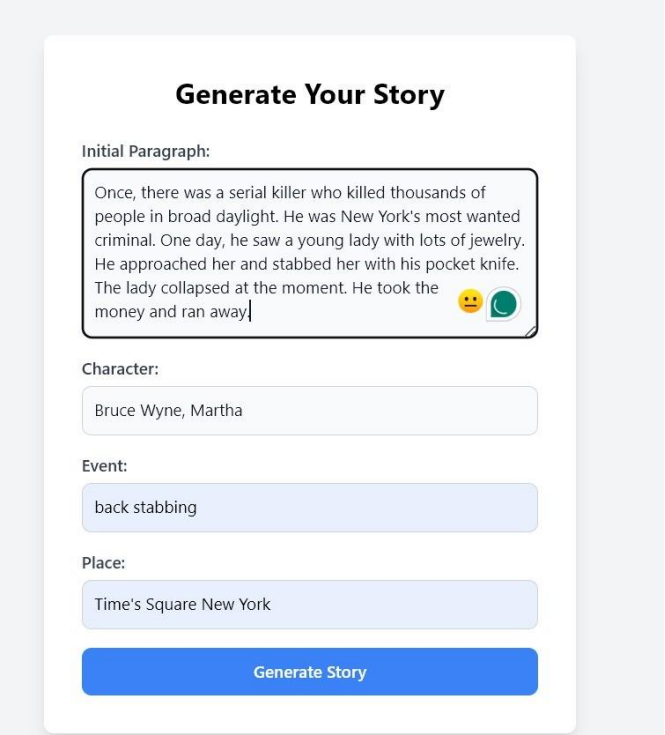


Fig. 2: UI of our story generator system

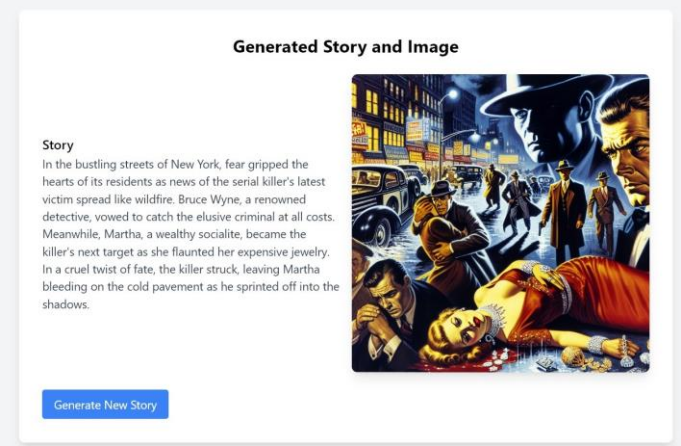


Fig. 3: First Generated para according to what Fig-2 gave

the story and the image (both paragraph-by-paragraph). The image in Fig-3 demonstrates one paragraph generated by the entities received from the front-end in Fig-2: In evaluating the quality of the generated story scenes, we employed an intrinsic evaluation method to assess the coherence of the generated paragraphs. Human evaluation was crucial to checking fluency, coherence, relevance, and likability. Fluency, for example, looks at how well-written individual sentences are, checking for things like proper grammar and clear meaning. We assessed each sentence on its own to ensure accuracy. Coherence is about the flow of ideas between paragraphs, making sure the story holds together smoothly. We wanted to see how well everything fits together to create a cohesive narrative. For relevance, we checked how closely the generated text matches the given topic or storyline. And lastly, likability measures how much readers enjoy the text and how engaging it is. Through these analysis, we ensured that the narratives we produce are logically connected and has users' immersion.

VI. Conclusion

Conclusively, we made a system that generates dynamic and multi-modal stories by combining fine-tuned GPT-3.5-turbo-0125 with Stable Diffusion for visual storytelling, generating narratives that are engaging and personalized with minimal human intervention, reflecting a significant step forward in the field of autonomous story generation. Looking ahead, we envision applications in gaming, education, and online content creation, where personalized and adaptive storytelling can add significant value. We anticipate AI models evolving to enhance coherence and narrative depth.

References

[1] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

[2] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., ... Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461.

[3] Fan, A., Lewis, M., Dauphin, Y. (2018). Hierarchical neural story generation. arXiv preprint arXiv:1805.04833.

[4] Kim, J., Heo, Y., Yu, H., Nang, J. (2023). A Multi-Modal Story Generation Framework with AI-Driven Storyline Guida nce. Electronics, 12(6), 1289.

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

[6] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

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

[8] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125, 1(2), 3.

[9] Akoury, N., Wang, S., Whiting, J., Hood, S., Peng, N., Iyyer, M. (2020). Storium: A dataset and evaluation platform for machine-in-the- loop story generation. arXiv preprint arXiv:2010.01717.

[10] stabilityai/sdxl-turbo· Hugging Face. (n.d.). https://huggingface.co/stabilityai/sdxl-turbo