Procedural Story Generation for Visual Novels Using Large Language Models and Text-to-Image Techniques

Dharma Hutama Husen
Department of Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
dharma.hutama@student.umn.ac.id

Wirawan Istiono
Department of Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
wirawan.istiono@umn.ac.id

Abstract*—Story-based video games have content limitations. A survey conducted among Genshin Impact players, 67% of respondents had reached the end of the avail- able story. The game's ending may lead to repetitive gameplay. To increase the story content inside video games, this research uses Generative AI for pro- ducing procedural stories. This research also measures the similarity of generated stories using a sentence similarity model. Text contents are generated using an LLM named mistralai/Mixtral-8x7B-Instruct-v0.1. Picture contents are generated using a Text-To-Image model named stabilityai/stable-diffusion- xl-base-1.0. Story similarities are measured using a sentence similarity model named sentence-transformers/all-MiniLM-L6-v2 with a similarity threshold of 73%. The visual novel application is implemented in Unity Engine 2022.3.26f1 version. The connection between Unity and the models are established through REST API provided by Hugging Face website. The result of this research is an implementation of a Visual Novel game with Generative AI-made content, as well as measuring the uniqueness level of 36,667% for low usage and up to 60% for high usage. The reduction of uniqueness level is attributed to the appearance of common genres, as well as the increasing number of comparisons required. This research shows how to implement Generative AI in a visual novel game, and shows that a Generative AI's ability to create unique stories will decrease with an increase in game repetitions.

***Keywords*— Large Language Models, Procedural Story Generation, Text-to-Image, Visual Novels**

I. INTRODUCTION

The video game industry has come a long way in recent years. Technological advancements have created more engaging experiences. However, game development still requires significant effort and time from developers, especially in creating diverse and unique content. This can be an obstacle for developers, especially those with limited resources [1], [2]. One type of video game that is gaining attention is story-based video games. Generally, these video games are in the Adventure or Role-playing genre. Based on a survey conducted by Statista in 2020 on America's favorite game genres, Adventure ranked fifth with 32% adoption while Role- Playing ranked seventh with 25% adoption. Story-based video games have low replayability, so players cannot continue the game when they run out of narrative content [3].

This research started from observations in Role-playing games (RPG) such as Genshin Impact and Warframe. There were several players who complained about the limited unique game content available after the main storyline. For example, an article from Cerafica explains that there is no game content for veteran players other than defeating Spiral Abyss or collecting character update materials. The remaining gameplay after the main storyline tends to be repetitive, even though some parts of the game are procedural. An example of a procedural system in Genshin Impact is the Mission system. The locations, enemy types, and mission types in the Commision system are procedurally generated. An example of a procedural system in Warframe is the map shape, as well as enemy spawning [4].

This research is based on a survey in an RPG game called Genshin Impact. This survey was conducted from May 8 to 12, 2024. The survey was distributed in the form of Google Forms to the Genshin Impact global community forum. There were 105 Genshin Impact player respondents, and there were three survey questions. The main survey question asked if the respondent had reached the end of the story. There were 67% of respondents who had reached the end of the story. This data proves that there are various veteran players who experience limited content in Genshin Impact.

To be a valid case study, it is necessary to prove that Genshin Impact has not used Generative AI (Artificial Intelligence) in the creation of its procedural content. These evidence are found through obser- vation. First, the production of Genshin Impact started from January 2017 and was published in September 2020. Meanwhile, ChatGPT, a chatbot that started the Large-Language-Model (LLM) trend, was published in November 2022 [5], [6]. Second, there is an article that examines the retelling of Genshin Impact in the form of derivative works. At section 2.1, it is explained that Genshin Impact's content is a framework for creating derivative works. This sentence implicitly proves that the game content, especially stories, are consistent be- tween different players [7], [8]. Lastly, the procedural content in Genshin Impact is not much different from the procedural content in older video games. For example, in Warframe, which has been published since March 25, 2013, exists a system that brings up various types of missions every day in various locations [9].

This research will be carried out in Visual Novel media because the addition of content in Visual Novel only requires text and images. Other game media, such as RPGs, require elements that are difficult to create through Generative AI, such as 3D models of enemies, location environments, and others. The procedural text of Visual Novel will be provided

by Large-Language-Model (LLM), while the procedural images will be provided by Text-To-Image model.

Generative AI is a new technology that has the potential to revolutionize the video game industry. Generative AI can be used to create game content procedurally, which means that content is created automatically by AI without human intervention [10]. This capability can help developers to create content more efficiently and diversely. Large-Language-Model and Text-To-Image models are included in Generative AI [11], [12]. By using Generative AI, a Visual Novel can provide a unique experience in the majority of play sessions. Writing a story starts from the premise, which is the basic idea that explains the story as a whole. Since story creation is based on the content of the premise, the uniqueness of the story can be measured by the uniqueness of the premise [13].

The premise similarity measurement method will be implemented based on a previous study named ”Performance of 4 Pre-Trained Sentence Transformer Models in the Semantic Query of a Systematic Review Dataset on Peri Implantitis” [14]. This research used various models and techniques to implement article search based on similarity in meaning. The measurement method is in the ”Semantic Text Similarity in the QT (Query Target) Corpus” section, where an FQ (Focus Question) measurement is performed on each QT that has been manually selected as an article related to the FQ. One of the models used is called sentence-transformers/all- MiniLM-L6-v2, this model gives an average of 73% in both test cases. This study will use that model and will set 73% as the similarity threshold, meaning that a pair of premises is designated as non-unique if the similarity score is equal to or greater than 73%.

II. LITERATURE STUDY

A. Visual Novel

Visual novel is a digital storytelling medium written in text. Visual novels have background images and character illustrations that change according to the situation and conditions of the story. Visual novels can also be supported by sound elements such as background music, sound effects, and voice acting of character dialog. Visual novels can provide interactive stories by giving choices to players. The storyline can change between players based on the choices given by each player. So that visual novels can provide varied stories until the end of the story. An example of branching storylines can be seen in the journal ”Designing A Visual Novel Game ”The Adventure Of Kabayan” As A Media For Studying English For Toefl”. Figure 1 illustrates a story tree taken from the journal, where the storyline develops from top to bottom. The branching of the story tree shows the story scenarios determined by the player’s choices [15].

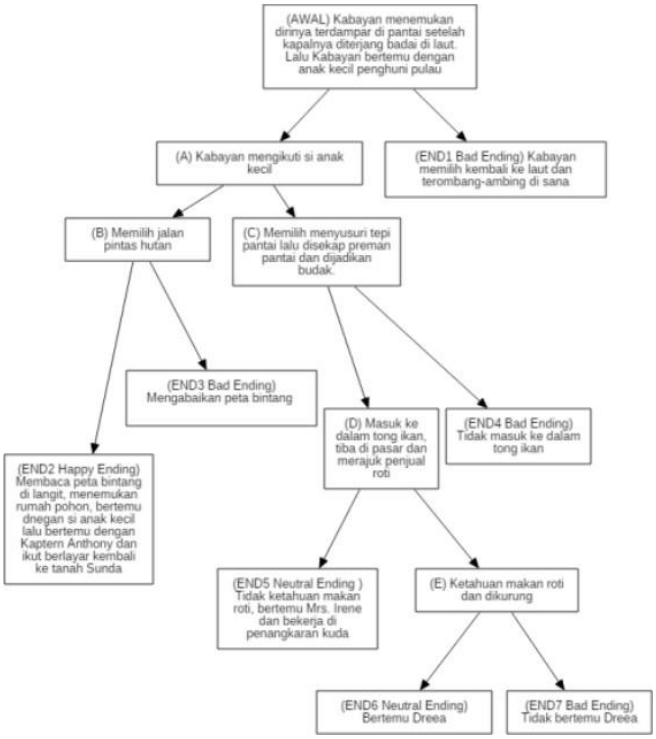


Fig. 1. Example of a story tree in a visual novel

B. Generative AI

Generative AI is a branch of artificial intelligence (AI) that focuses on creating content that appears new and meaningful, such as in the form of text, images, or sounds [16]. Generative AI is able to create new content based on patterns obtained in training data. Examples of generative AI are LLMs such as GPT, LLaMA, Gemma and text-to-image models such as Stable Diffusion [17], [18]. LLM creates new content by continuing a text work . While Stable Diffusion creates image content with a diffusion process guided by some keywords [19].

C. Large Language Model

A Language Model (LM) is a model that is trained to solve a language task such as translation, summarization, information retrieval, or conversational interaction. An LM that is capable of self-supervised training is called a Pre-trained Language Model (PLM). PLMs are capable of being trained on larger text datasets than LMs trained in a supervised environment. The performance in PLM easily increases based on the number of parameters in the model as well as the amount of training data. A Large Language Model (LLM) is a PLM with tens to hundreds of billions of parameters, and is trained using various Gigabytes to Terabytes of data. The transition from PLM to LLM occurs with the support of publishing Transformer architecture and the availability of large-scale training data [20].

There are different types of LLM models that are distinguished by the way the Attention mechanism is used in the Transformer as well as by the type of training applied. The initial process of training a model is referred to as pre-training, while subsequent training is referred to as fine-tuning. The main capabilities in an LLM model are determined by pre-training, e.g. the ability to create text based on a prefix is a type of Prefix Language Modeling pre-training, while the ability to complete text based on data before and after the location of the text completion target is a type of Masked

Language Modeling pre-training. After pre-training, an LLM model can create text, but for more specific uses the LLM model can go through a fine-tuning process. Examples of fine-tuning are Instruction-tuning and Alignment-tuning. Instruction-tuning can help LLM to follow an instruction, while Alignment-tuning is done to reduce malicious text output [21]. The LLM training process is illustrated in Figure 2. RL stands for Reinforcement Learning, RM stands for Reward Modeling, RLHF stands for Reinforcement Learning with Human Feedback.

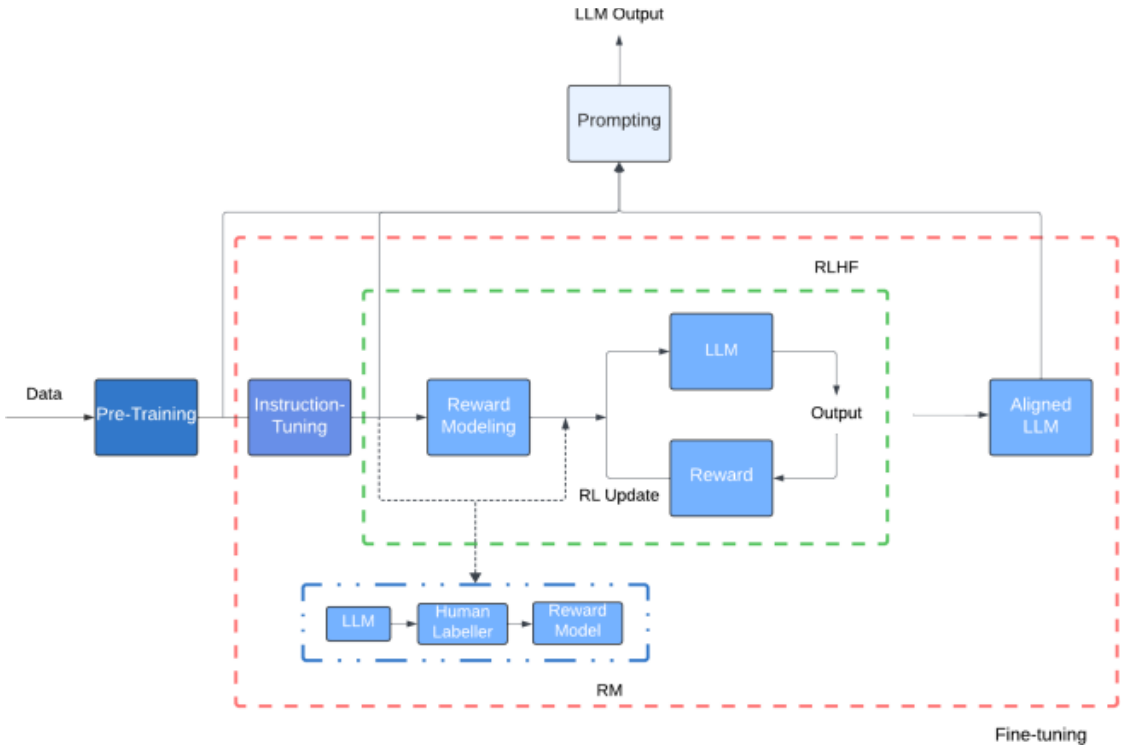


Fig. 2. LLM training process

D. Transformer

Transformers are a type of neural network architecture. Transformers are commonly used in LLMs due to their ability to process data in parallel. In the context of text, Recurrent Neural Networks (RNN) analyze text elements one-by-one, while Transformers can break text elements into various tokens that can be processed at once by the Attention mechanism to get the next token prediction. The Transformer architecture breaks data into tokens to be processed in parallel, this parallelism makes LLMs scalable [22].

The implementation of Transformer in various models may vary. These variations are named based on the part of the Transformer used, such as encoder-decoder model or decoder-only model. Encoding is the process of converting data into tokens, while decoding is the process of converting tokens into data [23].

E. Token

A token, in the context of Transformer, is a high-dimensional vector of numbers. Each token in Transformer has an abstract meaning consisting of individual numbers. A pair of tokens can have a positive or negative correlation based on the dot product of the two vectors. A set of tokens can be represented by a matrix, while the operations performed by the Encoding, Decoding, and Attention mechanisms can be represented by matrix multiplication using the matrix of the set of tokens [24].

F. Encoding and Decoding

The Encoding process is the process of converting data into Tokens, and the Decoding process is the opposite. The data in question can be text, images, sounds, and others. The tokenized data produces tokens related to the type of data used, for example, for text data, the token is a piece of text, for image data, the token is a piece of image, and so on [25].

All possible tokens used by the model are specified at the beginning of modeling, a matrix token ”dictionary” that is used by the encoder and decoder as the basis for tokenization. For example, in the context of text, a token can represent a letter, a word fragment, a whole word, a number, or a symbol. The numbers in the encoder matrix and decoder matrix are determined after a training process [26].

G. Attention

Tokens that have gone through the tokenization process enter the Attention mechanism process. The Attention mechanism is a process that is repeated over a set of tokens, to distribute token context information to every other token. This process is a matrix multiplication between the token matrix and a weight matrix created from the training process. Through the Attention process, a token that only represents a piece of data can absorb context from other pieces of data [27].

There are two types of context absorption processes that can be trained in the Attention mechanism, namely Causal Decoder and Prefix Decoder. In Causal Decoder, each token has an order, and the context absorption of each token only occurs from the token with the previous order. Causal Decoder

is useful for creating models that can create text sequentially. In Prefix Decoder or also referred to as Non-causal Decoder, context is absorbed bidirectionally. Prefix Decoder is useful for creating models that can complete an image or document using context data before and after the truncated part [28].

H. Prompt Engineering

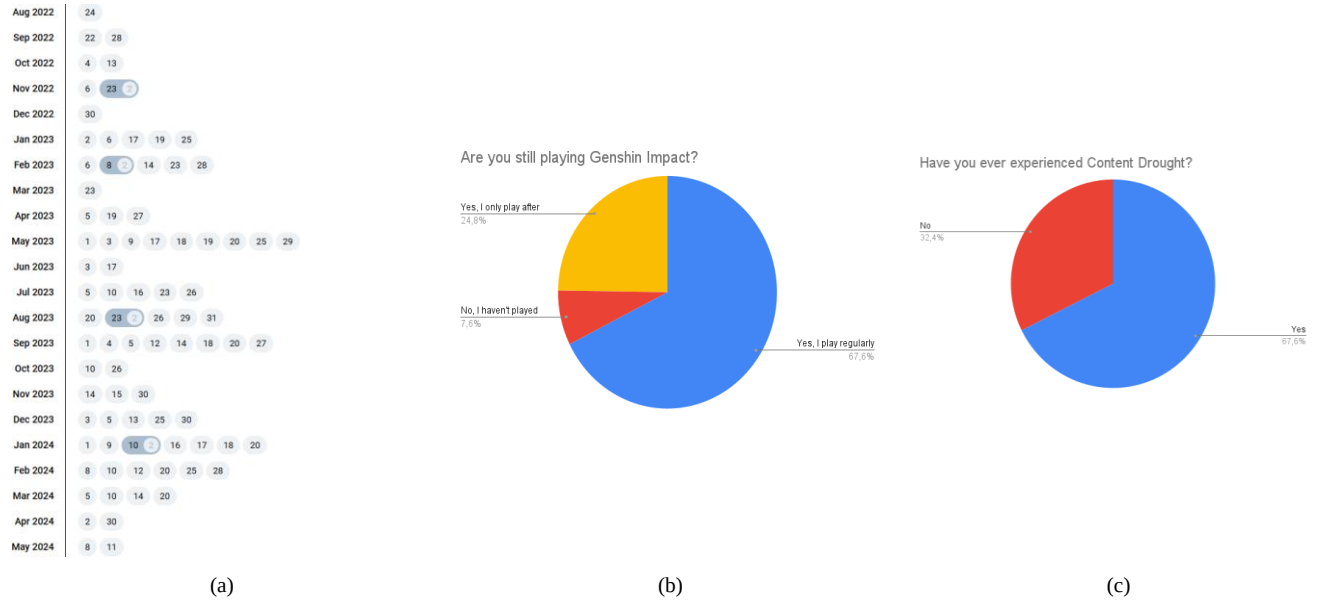
Prompt engineering is a technique of using the Language Model using prompt patterns that can add to the Language Model's ability to complete a particular task. Prompt engineering can be used to mitigate LLM limitations, such as: lack of strong recall, inconsistent output from the same input, and fabrication of incorrect content such as hallucinations [29].

One example of prompt engineering is prompt chaining, which means dividing a task into various sub-tasks, then the output of one sub-task is used as input for another sub-task. Prompt chaining can make LLM more transparent and reliable, and facilitate analysis if LLM does not successfully perform its task [30].

III. METHOD

A. Problem Identification

The problem was identified by conducting a survey on an RPG game called Genshin Impact. This survey was conducted from May 8 to 12, 2024. The survey was distributed in the form of Google Forms to the global forum of the Genshin Impact community. Each respondent's email may only provide one response. There were 105 Genshin Impact player respondents, and there were three survey questions. The first question asked for the date the respondent started playing Genshin Impact. This question was necessary because Genshin Impact games can change over time. Older dates represent players who played earlier version of Genshin Impact with less content variety. 74% of respondents started playing Genshin Impact after January 1, 2023, while Genshin Impact has been published since September 28, 2020. The second question asked if the respondent are still actively playing Genshin Impact. This question is needed because the frequency of playing can determine the speed at which players reach the end of the story. There are 67% of respondents who are still actively playing Genshin Impact, there are 24% of respondents who play when there is new content, and the rest are respondents who have not played Genshin Impact. The third question asked if the respondent had reached the end of the story. There are 67% of respondents who have reached the end of the story. The result of each survey questions has been illustrated in Figure 3.

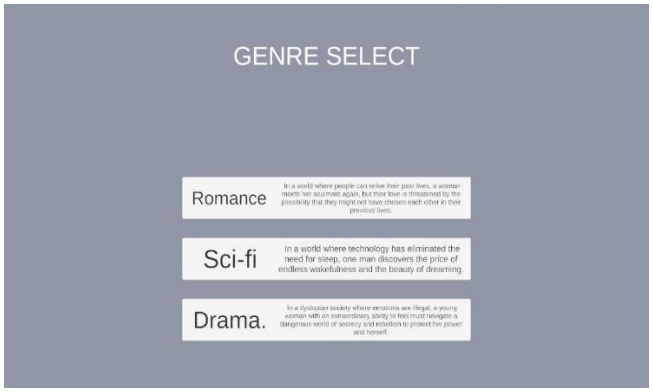


(a) (b) (c)

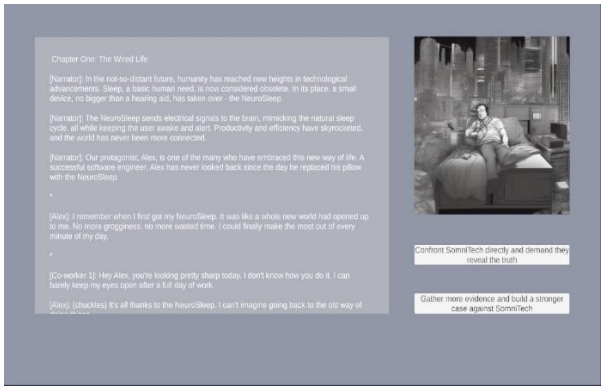
Fig. 3. The result of the (a) first survey question, (b) second survey question, and (c) third survey question

B. Prototyping

The visual novel application will be divided into two main sections namely the Story Genre Selection section and the Story Content Creation section. In the Story Genre Selection section, the application provides a choice of story genres to be created. Once a Genre is selected, the application runs the Story Content Creation section. In the Story Content Creation section, the app provides story content in the form of text and images. Figure 4(a) shows the user interface for the Story Genre Selection section, and Figure 4(b) shows the user interface for the Story Content Creation section.



(a)



(b)

Fig. 4. The user interface for (a) the Story Genre Selection and (b) the Story Genre Selection section

Before story creation, the player will be given a choice of Genre of story to produce. At the beginning of the app opening, LLMs are assigned to provide three genre ideas for short stories. After that, each genre is given to LLMs with the task of creating an interesting premise of one sentence length. Once all the premises have been created, the application moves from the loading page to the genre selection page.

After the player selects the genre, the story is slowly created. Starting from creating an outline of the contents of each chapter. Then the first chapter begins to be written, supporting images are created, and at the end of the story two options can be provided. Once a continuation of the story is selected, the outline of the next chapter is updated with the desired storyline. Then the story creation process repeats for the next chapter, until there is no continuation. If there are no continuations, this session's story creation process is complete.

The process of creating supporting images occurs after the story in the chapter has been written. LLM will analyze and then describe visual elements that can support the storyline. This visual element description is given to Stable Diffusion, then the Stable Diffusion result image is displayed before the story page is displayed.

Every use of Generative AI for text or image output has the same process flow. First, a prompt is created and parameters are set. Then the local application makes a POST request with the contents of the previous data, and uses the link model and API key. The POST request will be replied to with the result of the data made by Generative AI. This data is in the form of JSON, so there is a process of separating the data so that it can be used by the GameObject component in Unity.

C. Implementation

This research uses a Generative AI LLM named *mistralai/Mixtral-8x7B-Instruct-v0.1* and an AI Text- To-Image model named *stabilityai/stable-diffusion-xl-base-1.0*, both hosted in the cloud on the Hugging Face website. The application is implemented in Unity Engine version 2022.3.26f1. The sentence similarity model used to measure premise similarity is named *sentence-transformers/all-MiniLM-L6-v2*. Access between the Unity application and the Hugging Face website is implemented through the REST API.

There are several types of prompts in this application system. These sentences form the basis of the prompt creation process. Some prompts require variable data, variable data is written as "(variableName)". In the prompt creation process, the basic prompt sentence will be filled with the specified variable data.

D. Genre Creation Prompt

This prompt is used to create a story genre. The output is "(genre),(genre),(genre)". This prompt is used when the application is first opened. Here are the contents of the prompt: "*Suggest 3 story genre that is suitable for a short story. Answer as a comma-separated string. Do not give explanations".*

E. Premise Creation Prompt

This prompt is used to create the premise of a genre. The output is "(premise)". This prompt is used in each of the available genres. Here are the contents of the prompt:

Create a summary for a short story with the given genre. The summary must be able to hook readers to start reading the story. The summary must be one sentence long. Avoid creating cliche stories. For example: "forbidden love", "memory loss", "murder mystery", these kinds of stories are too saturated.

Genre: (genre)

F. Story Outline Creation Prompt

This prompt is used to create a story outline as a direction for the story to be created. The output is "(storyOutline)". This prompt is used after the player has selected the genre and premise. The selected genre and premise will be part of this prompt. Here are the contents of the prompt:

You're an aspiring novel writer that is eager on creating a new story. You will be given the story genre and the target summary. Create the outline of such story.

Genre: (genre) Summary: (premise)

G. Chapter Writing Prompt

This prompt is used to create a story chapter. The output is "(storyChapter) !!!Choice!!! #Op- tion#(selectedContinuation option) #Option# (selectedContinuation option)". The "!!!Choice!!!" and "#Op- tion#" text formats are used to facilitate text processing, as the story must be separated from the option text. The genre and story outline will be used to direct the writing of the story and options. Here are the contents of the prompt:

You are a Visual Novel. Create chapter one for the given genre and the entire story outline. Finish the Chapter, and give the player some choice. While the player is near other characters, use internal monologues sparingly, instead expose information through dialogue between those characters. Make each character's speech easily distinguishable. A character speech, including the narrator, must use the following format:

[Character Name]: "Message"

For example: [Narrator]: "Once upon a time..."

Except for chapter title and choices, the story should exclusively consist of character speeches. The chapter MUST end in an open-ended choice for the player as the transition for the next chapter. Use this example:

!!!Choice!!!

#Option# [describe option here] #Option# [describe option here] Genre: (genre)

Outline: (storyOutline)

H. Visual Summary Prompt

This prompt is used to summarize the visual elements of the story that can be used by Stable Diffusion.

The story in the chapter will be used as part of this prompt. Here are the contents of the prompt:

You will be given a part of a story in a Visual Novel. Create a stable diffusion prompt for creating an image that supports the narrative. A stable diffusion prompt is a collection of keywords about what should be drawn on the novel background. While creating the prompt, avoid asking for text, dialogue boxes, and anything that would contaminate the background picture. The picture must be mostly scenery. Put characters in the sides or corner of the image.

Story: (storyChapter)

I. Story Outline Update Prompt

This prompt is used to update the story outline. The story outline can change drastically because the options given might be contradictory. This prompt is used after the player has made a story continuation choice. The contents of the story choice, the last story outline, and the last chapter story will be used as part of this prompt. Here are the contents of the prompt:

The Current story Outline may be outdated. Please update it with the newest information from the Current Story and the Chosen player Action. ONLY OUTPUT THE NEW OUTLINE, do not output anything else.

Chosen Action: (chosenStoryContinuation)

Current Outline: (currentStoryOutline)

Current Story: (storyChapter)

J. Continued Chapter Writing Prompt

This prompt is similar to the first story creation prompt. The difference is that this prompt should generate the next chapter of the story. The genre and outline of the latest story will be used to direct the follow-up chapter. Here are the contents of the prompt:

You are a Visual Novel. I will give you the story so far and the entire story outline, your task is to complete the next chapter. ONLY OUTPUT THE NEXT CHAPTER, do not repeat the current story....

The new chapter MUST end in a new choice for the player as the transition for the next chapter. DO NOT append the last player choice here. Use this format:

!!!Choice!!!

#Option# [describe option here]

#Option# [describe option here]

Chosen Action: (chosenStoryContinuation)

Outline: (newStoryOutline)

K. Testing

In the test, the system workflow was changed to allow the application to create various consecutive stories. The genre data and premises are stored in CSV files, The testing version does not use stable diffusion. To obtain accurate measurements, the prompts used in the test were not changed from the prompts used by normal use. When the application is opened normally, there are three genre options. These three genres are always unique because they originate from a single prompt. The prompts can easily be changed to create more genres at once, but to simulate normal usage, the genres were kept at three at a time, but repeated as many times as the test scenario required.

L. Measuring Premise Similarity Using Sentence Similarity

The first part of testing is the premise test data creation. The prompt starts with genre creation, but a single prompt process only generates three genres. To simulate repeated use, the prompt will be used repeatedly and the genre results are collected for the premise generation process. The test will simulate 100 times of application usage.

Each genre is created as well as its premise data using the premise generation prompt. In normal usage only one of the three genres is selected, but each premise content is still displayed when opening the app. Uniqueness is still measured for each premise displayed, so there will be a total of 300 premise data.

After collecting the test data, the similarity of each premises will be measured using the sentence similarity model. The sentence similarity model used in this research is sentence-transformers/all-MiniLM-L6- v2 through the REST API provided by Hugging Face. Each available premise will be measured for similarity with every other premise. The similarity result is between 0 and 1, low numbers indicate unique premises, while high numbers indicate repeating premises.

In the practice of using sentence similarity, the 300 premises that have been created will be entered into the list. Then the source sentence parameter will be filled by the first premise, while the sentences parameter will be filled by the list of premises. The sentence similarity model will give a similarity score between the source sentence and each sentences element, this score will be recorded per column in a CSV table row. The process is then repeated until the last premise has been assigned as the source sentence.

This process generates a 300x300 number table. The table data is saved in CSV file format to facilitate analysis, graphing, and inclusion of attachments. The data is stored in

an excel sheet starting from cell A2 to cell KN301. From the 300 premises available, scenarios of using 50, 100, 200, and 300 premises can be created to compare changes in uniqueness.

Measuring Premise Uniqueness

The sentence similarity model gives a continuous value between 0 and 1 in the comparison between two texts, but in this research it is necessary to compare the uniqueness of a text with many texts. The expected type of uniqueness determination is also different from the available data, for example, the data can prove if the two texts are 57% similar, but the number cannot show if there has been a similar text before. Therefore, some additional processing is required before reaching the uniqueness calculation result.

It is necessary to set a uniqueness threshold that determines if a text is unique or not. The uniqueness threshold is a number between 0% and 100%, indicating the level of tolerance for similarity. For example, if there are two texts with a score of 57% based on sentence similarity, with a uniqueness threshold of 40% these two texts are not considered unique, but with a uniqueness threshold of 60% these two texts are still considered quite unique. In this research, the uniqueness threshold is set as 73% based on previous research [9].

The uniqueness of the premises will be measured based on the previously observed premises. The first premise observation is always set as the unique premise, as there are no previous premise observations to compare with. The 50th premise observation will be compared with the previous 49 premises, if there is a score that exceeds the uniqueness limit, this premise is designated as non-unique.

TABLE I. UNIQUENESS FORMULA PLACEMENTS

1&1 Similarity	1&2 Similarity	-	Code A	Code B
2&1 Similarity	2&2 Similarity	-	Code C	-
-	-	-	-	-
-	-	-	Code D	-

To perform these calculations in the excel sheet, two types of excel code snippets are required with the placement as shown in Table 1. Codes A, B, and C are the first type of code, while code D is the second type of code. The first type of code is responsible for calculating the number of scores that exceed the uniqueness limit. The difference between code A and code B is the uniqueness limit number, this piece of code continues to the right until it reaches 100% uniqueness limit. The difference between code A and code C is the number of scores counted, code A only counts the first premise score while code B counts the first and second premise scores, this code snippet continues downwards until it reaches the bottom row of the premise data. The value of each of these codes is reduced by one to ignore the self-counting of the premise scores that always occurs on the diagonal of the data table. The second type of code is responsible for counting the number of each row above it that has a value of 0. This means the number of premises that, when observed with each previous premise, never exceed the uniqueness limit. This code changes for different premise usage scenarios, e.g. scenario 300 uses range KS2:KS301, while scenario 100 uses range KS2:KS101. The code continues to the right to calculate each uniqueness constraint from 10% to 100%. Here are the contents of each Code:

Code A: =SUM(IF($A2:$A2¿=0.1;1;0)) - 1

Code B: =SUM(IF($A2:$A2¿=0.2;1;0)) - 1

Code C: =SUM(IF($A3:$B3¿=0.1;1;0)) - 1

Code D: =SUM(IF(KS2:KS301=0;1;0))

IV. RESULTS AND DISCUSSION

The Excel sheet at the end of testing has an area of 300*300 which is the premise similarity data. On the right, there is a 10*300 area which is the calculation of the number of premises that have a score higher than the uniqueness limit of each column. Then there is a 10*1 area that contains the number of unique premises for each uniqueness threshold in a scenario. The number of unique premises is then divided by the number of premises used to get the percentage of premise uniqueness. The percentage of uniqueness for each testing scenario are illustrated in Figure 5. The x-axis represents the uniqueness threshold, if there is a premise with a similarity score higher or equal to the uniqueness threshold, then the premise is determined as non-unique for that uniqueness threshold. The y-axis represents the percentage of premises that are unique compared to all previous premises. Each line shows a scenario of premise usage.

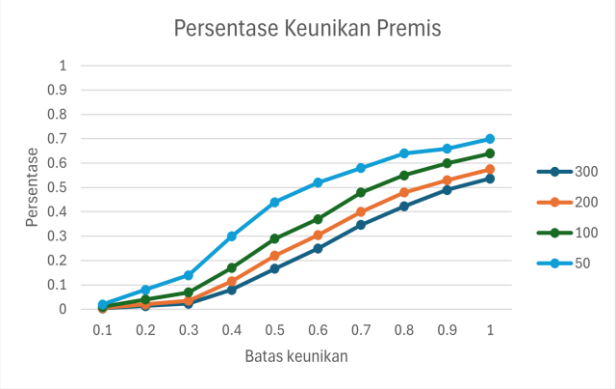


Fig. 5. The percentage of uniqueness for each testing scenario

The graph data that shown in Figure 5 shows that there are several premises that are written exactly the same. The proof is found in the 100% uniqueness limit column. If there is even the slightest difference between each pair of texts, then the similarity score of that pair of texts is below 100%, so the value for the 100% uniqueness threshold should have been 100% unique.

The graph data also shows that the more the number of application uses increases, the rarer the exis- tence of a new unique premise. This happens because in low usage, almost every premise created is new. How- ever, with continued usage, similar premises can reappear, thus reducing the uniqueness rate. At the uniqueness threshold set as 73%, the uniqueness value in the 50 premise scenario is 60%, the uniqueness value in the 100 premise scenario is 49%, the uniqueness value in the 200 premise scenario is 42%, and the uniqueness value in the 300 premise scenario is 36.667

The decrease in uniqueness percentage can also be explained by the number of occurrences of each genre in the test data. There are certain genres that are more likely to appear than others. For example, genres such as Romance and

Science Fiction are more likely to appear compared to genres such as Drama or Comedy. Genre occurrence frequency are illustrated in Figure 6.

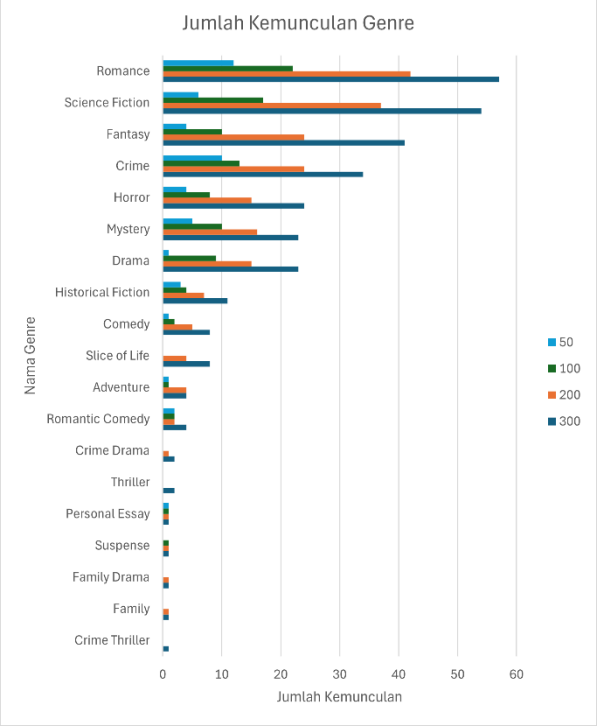


Fig. 6. Genre occurrence frequency in each testing data scenario

CONCLUSION

Visual Novel game with procedural content made by Generative AI has been implemented. The level of uniqueness of the story made by Generative AI based on sentence similarity is determined by the number of Visual Novel game usage and the uniqueness tolerance limit number. In using less than 50 premises, the uniqueness rate tends to be above 60%. While in the use of more than 300 premises, the uniqueness rate tends to be below 36.667%.

ACKNOWLEDGMENT

Thank you to Universitas Multimedia Nusantara in Indonesia for providing a space for academics to conduct this journal research. Hopefully, this research will contribute significantly to the growth of technology in Indonesia.

REFERENCES

[1] C. Politowski, F. Petrillo, G. C. Ullmann, and Y. G. Guéhéneuc, "Game industry problems: An extensive analysis of the gray literature," *Inf. Softw. Technol.*, vol. 134, 2021, doi: 10.1016/j.infsof.2021.106538.

[2] G. Freeman, N. McNeese, J. Bardzell, and S. Bardzell, "'Pro-amateur'-driven technological innovation: Participation and challenges in indie game development," *Proc. ACM Human-Computer Interact.*, vol. 4, no. GROUP, 2020, doi: 10.1145/3375184.

[3] Anna Fleck, "Americans' Favorite Video Game Genres," statista.com. Accessed: Jul. 18, 2024. [Online]. Available: https://www.statista.com/chart/24700/favored-video-game-genres-in-the-us/

[4] C. Anna, "Genshin Impact Needs to Address its Problem With Content Droughts," gamerant.com. Accessed: Nov. 17, 2024. [Online]. Available: https://gamerant.com/genshin-impact-content-droughts/

[5] R. Xu, Y. Huang, X. Chen, and L. Zhang, "Specializing Small Language Models Towards Complex Style Transfer via Latent Attribute Pre-Training," *Front. Artif. Intell. Appl.*, vol. 372, no. 3, pp. 2802–2809, 2023, doi: 10.3233/FAIA230591.

[6] X. Peng, T. Liu, and Y. Wang, "Genshin: General Shield for Natural Language Processing with Large Language Models," 2024, [Online]. Available: http://arxiv.org/abs/2405.18741

[7] M. Greting, X. Mao, and M. P. Eladhari, "What Inspires Retellings - A Study of the Game Genshin Impact," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 13762 LNCS, pp. 249–269, 2022, doi: 10.1007/978-3-031-22298-6_16.

[8] C. Li *et al.*, "ChatHaruhi: Reviving Anime Character in Reality via Large Language Model," 2023, [Online]. Available: http://arxiv.org/abs/2308.09597

[9] D. Ronaldo and C. H. Karjo, "A semantic study of 'warfare' video game terminologies," *AIP Conf. Proc.*, vol. 2594, no. January, 2023, doi: 10.1063/5.0109937.

[10] U. Mittal, S. Sai, V. Chamola, and Devika, "A Comprehensive Review on Generative AI for Education," *IEEE Access*, no. August, 2024, doi: 10.1109/ACCESS.2024.3468368.

[11] J. Lee, S. Eom, and J. Lee, "Empowering Game Designers With Generative Ai," *Iadis Int. J. Comput. Sci. Inf. Syst.*, vol. 18, no. 2, pp. 213–230, 2023, doi: 10.33965/ijcsis_2023180213.

[12] Y. Sun, Z. Li, K. Fang, C. H. Lee, and A. Asadipour, "Language as Reality: A Co-creative Storytelling Game Experience in 1001 Nights Using Generative AI," *Proc. - AAAI Artif. Intell. Interact. Digit. Entertain. Conf. AIIDE*, vol. 19, no. 1, pp. 425–434, 2023, doi: 10.1609/aiide.v19i1.27539.

[13] M. Shidiq, "the Use of Artificial Intelligence-Based Chat-Gpt and Its Challenges for the World of Education; From the Viewpoint of the Development of Creative Writing Skills," *Soc. Humanit.*, vol. 01, no. 01, p. 2023, 2023.

[14] C. Galli, N. Donos, and E. Calciolari, "Performance of 4 Pre-Trained Sentence Transformer Models in the Semantic Query of a Systematic Review Dataset on Peri-Implantitis," *Inf.*, vol. 15, no. 2, 2024, doi: 10.3390/info15020068.

[15] [15] R. Kurniawan and G. N. Windari, "Perancangan Game Visual Novel ' the Adventure of Kabayan ' Sebagai Media Belajar Bahasa Inggris Untuk Toefl," *J. Masy. Inform. Indones.*, vol. 4, no. 28, pp. 5–10, 2019.

[16] A. Bandi, P. V. S. R. Adapa, and Y. E. V. P. K. Kuchi, "The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges," *Futur. Internet*, vol. 15, no. 8, 2023, doi: 10.3390/fi15080260.

[17] F. Fui-Hoon Nah, R. Zheng, J. Cai, K. Siau, and L. Chen, "Generative AI and ChatGPT: Applications, challenges, and AI-human collaboration," *J. Inf. Technol. Case Appl. Res.*, vol. 25, no. 3, pp. 277–304, 2023, doi: 10.1080/15228053.2023.2233814.

[18] F. J. García-Peñalvo and A. Vázquez-Ingelmo, "What Do We Mean by GenAI? A Systematic Mapping of The Evolution, Trends, and Techniques Involved in Generative AI," *Int. J. Interact. Multimed. Artif. Intell.*, vol. 8, no. 4, pp. 7–16, 2023, doi: 10.9781/ijimai.2023.07.006.

[19] M. Muller, L. B. Chilton, A. Kantosalo, M. Lou Maher, C. P. Martin, and G. Walsh, "GenAICHI: Generative AI and HCI," *Conf. Hum. Factors Comput. Syst. - Proc.*, 2022, doi: 10.1145/3491101.3503719.

[20] M. Usman Hadi *et al.*, "Large Language Models: A Comprehensive Survey of its Applications, Challenges, Limitations, and Future Prospects," *Authorea Prepr.*, 2024, [Online]. Available: https://www.authorea.com/users/618307/articles/682263-large-language-models-a-comprehensive-survey-of-its-applications-challenges-limitations-and-future-prospects

[21] C. Xu and J. McAuley, "A Survey on Model Compression and Acceleration for Pretrained Language Models," *Proc. 37th AAAI Conf. Artif. Intell. AAAI 2023*, vol. 37, pp. 10566–10575, 2023, doi: 10.1609/aaai.v37i9.26255.

[22] M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy, "Do Vision Transformers See Like Convolutional Neural Networks?," *Adv. Neural Inf. Process. Syst.*, vol. 15, no. NeurIPS, pp. 12116–12128, 2021.

[23] J. Maurício, I. Domingues, and J. Bernardino, "Comparing Vision Transformers and Convolutional Neural Networks for Image Classification: A Literature Review," *Appl. Sci.*, vol. 13, no. 9, 2023, doi: 10.3390/app13095521.

[24] L. Xu, W. Ouyang, M. Bennamoun, F. Boussaid, and D. Xu, "Multi-class Token Transformer for Weakly Supervised Semantic Segmentation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern*

Recognit., vol. 2022-June, pp. 4300–4309, 2022, doi: 10.1109/CVPR52688.2022.00427.

[25] T. Nayak and H. T. Ng, "Effective modeling of encoder-decoder architecture for joint entity and relation extraction," *AAAI 2020 - 34th AAAI Conf. Artif. Intell.*, pp. 8528–8535, 2020, doi: 10.1609/aaai.v34i05.6374.

[26] J. Ning *et al.*, "All in Tokens: Unifying Output Space of Visual Tasks via Soft Token," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 19843–19853, 2023, doi: 10.1109/ICCV51070.2023.01822.

[27] M. S. Ryoo, A. J. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova, "TokenLearner: Adaptive Space-Time Tokenization for Videos," *Adv. Neural Inf. Process. Syst.*, vol. 16, no. NeurIPS, pp. 12786–12797, 2021.

[28] J. Seo, S. Lee, L. Liu, and W. Choi, "TA-SBERT: Token Attention Sentence-BERT for Improving Sentence Representation," *IEEE Access*, vol. 10, pp. 39119–39128, 2022, doi: 10.1109/ACCESS.2022.3164769.

[29] A. Gao, *Prompt Engineering for Large Language Models*. Springer Nature Singapore, 2023. doi: 10.2139/ssrn.4504303.

[30] S. Lim and R. Schmälzle, "Artificial intelligence for health message generation: an empirical study using a large language model (LLM) and prompt engineering," *Front. Commun.*, vol. 8, 2023, doi: 10.3389/fcomm.2023.1129082.