MarioGPT: Open-Ended Text2Level Generation through Large Language Models

Viet Nhat Nguyen (21520378)

Thi Lien Le (21522282)

July 9, 2023

UNIVERSITY OF INFORMATION TECHNOLOGY

Table of Contents

1 Introduction

▶ Introduction

▶ Background and Related Work

▶ Open-Ended Level Generation through LLMs

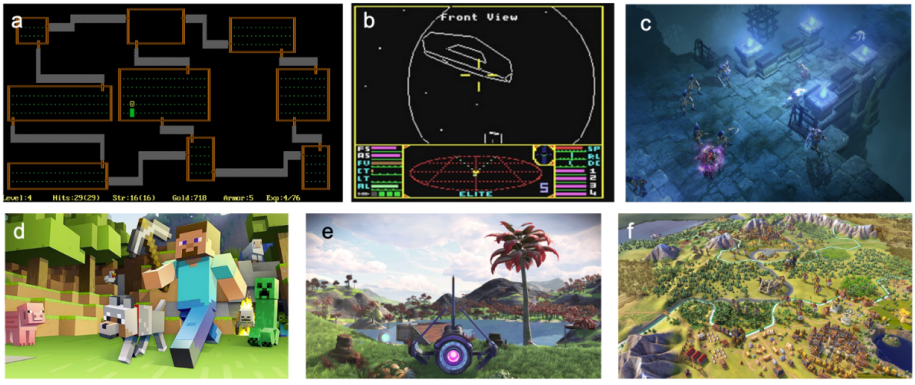
▶ Experiments and Results

▶ Conclusion

Procedural Content Generation (PCG)

- Procedural Content Generation (PCG) refers to techniques that can automatically create game content (e.g. levels, maps, or characters).
  - Increasing the replayability
  - Reducing production costs

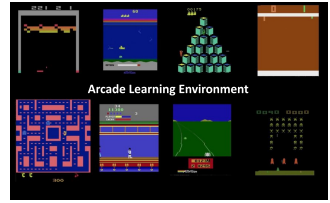
Procedural Content Generation (PCG)

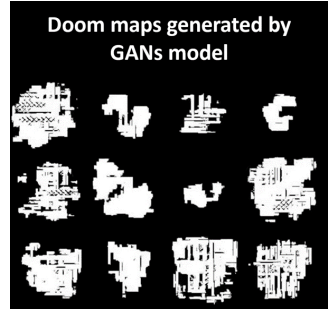


a) Rouge **b**) Elite **c**) Diablo III **d**) Minecraft **e**) No Man's Sky **f**) Civilisation VI

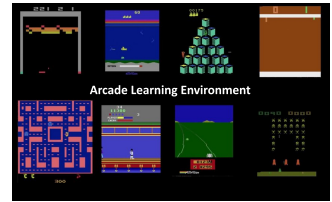
PCG and Machine Learning

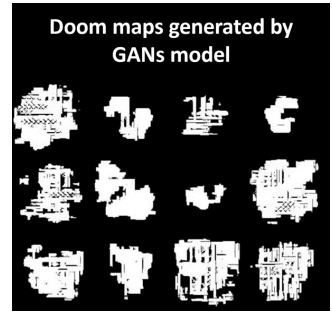
1 Introduction





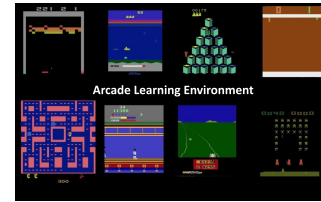
- PCG for evaluating generalization capabilities of trained agents.
  - Offer more diverse and challenging scenarios.
  - A better way to test the adaption and generalization of trained agents.

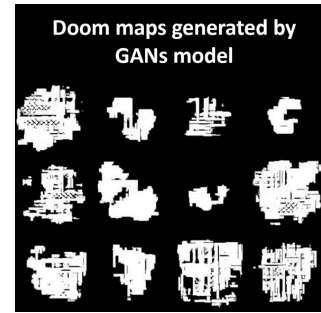




- PCG for evaluating generalization capabilities of trained agents.
  - Offer more diverse and challenging scenarios.
  - A better way to test the adaption and generalization of trained agents.

- Incorporation of machine learning-based approaches in PCG systems.
  - GANs for generating levels (Doom, Super Mario Bros)





Challenges

- **Costly** searching in the latent space of neural networks.
- Desire to **directly condition** a generator for creating levels with specific properties, ideally in natural language. (**prompt**)

Contributions

1. Introduction of MarioGPT, a text-to-level model generating Mario levels based on natural language prompts.

Contributions

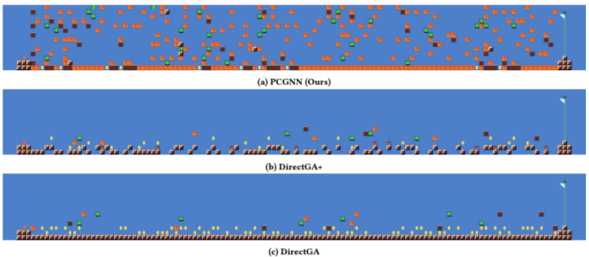
1. Introduction of MarioGPT, a text-to-level model generating Mario levels based on natural language prompts.
2. Combining MarioGPT with novelty search for producing diverse levels.

Table of Contents

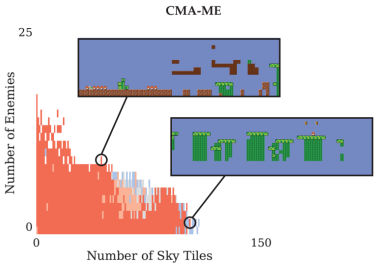
2 Background and Related Work

- Introduction
- Background and Related Work
- Open-Ended Level Generation through LLMs
- Experiments and Results
- Conclusion

Neural Network-based Level Generation

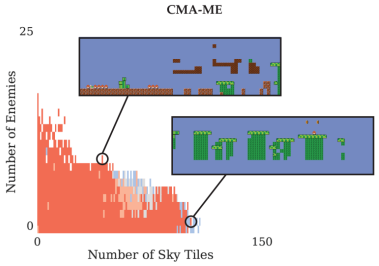


Using neural networks (Beukman et al.[1])



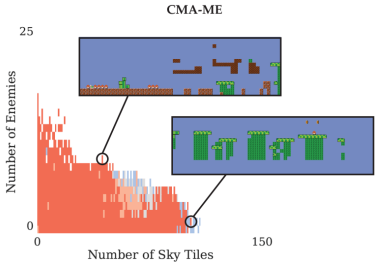
Searching in the latent space of GANs (Matthew et al.[2])

- Guided sampling of the latent space could result in a diverse set of levels.
- The abilities to control characteristics in generated levels.



Searching in the latent space of GANs (Matthew et al.[2])

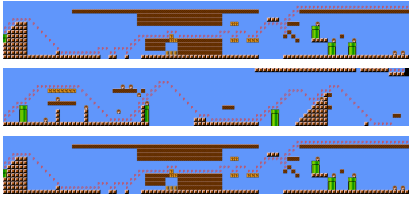
- Guided sampling of the latent space could result in a diverse set of levels.
- The abilities to control characteristics in generated levels.
- *Using A* agent to measure whether the generated levels were playable.*



Searching in the latent space of GANs (Matthew et al.[2])

Open-Endedness Paradigm

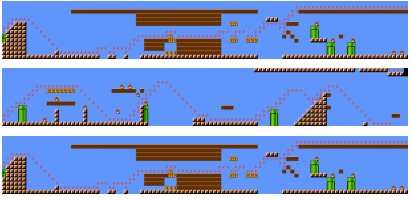
- The open-endedness paradigm focuses on algorithms that can produce infinite innovation.



Diversity but not-playable cases

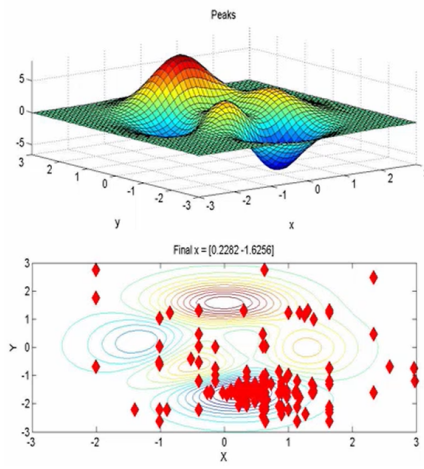
Open-Endedness Paradigm

- The open-endedness paradigm focuses on algorithms that can produce infinite innovation.
- *Must balance the hard task of generating content with **diversity** as well as **playability***



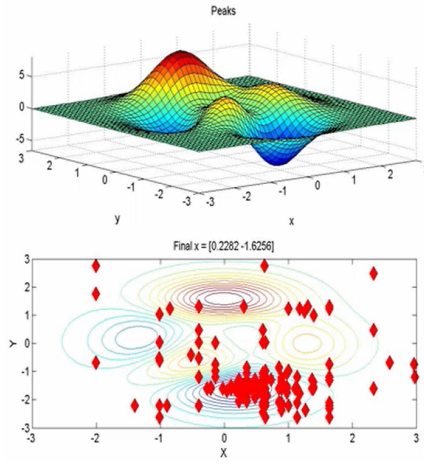
Diversity but not-playable cases

- Uses concepts from *evolutionary biology*

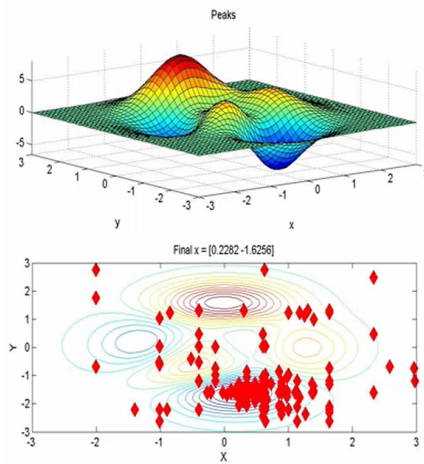


Genetic Algorithms

- Uses concepts from *evolutionary biology*
- Start with an initial generation of candidate solutions that are tested against the objective function

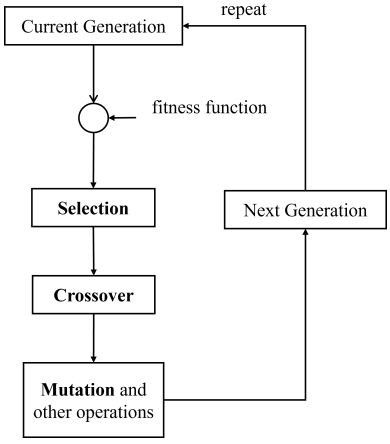


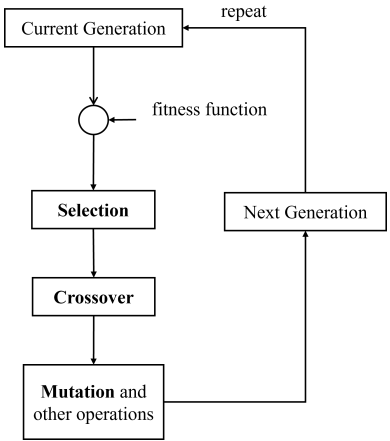
- Uses concepts from *evolutionary biology*
- Start with an initial generation of candidate solutions that are tested against the objective function
- Subsequent generations evolve from the 1st through *selection*, *crossover* and *mutation*



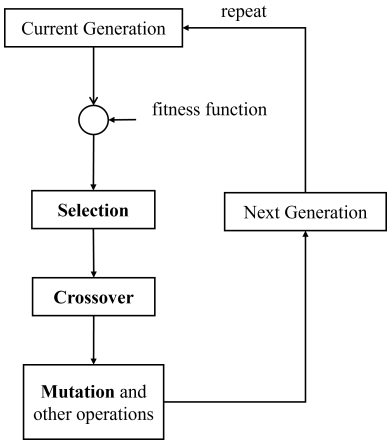
Genetic Algorithms

2 Background and Related Work

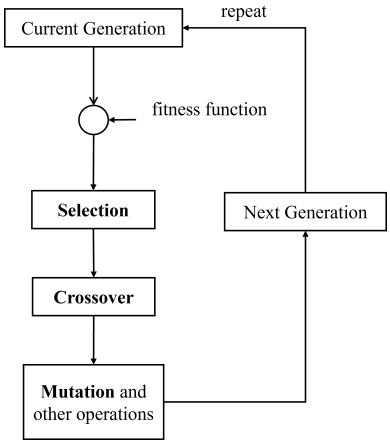




- First generation is arbitrarily, or $x = random()$



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*

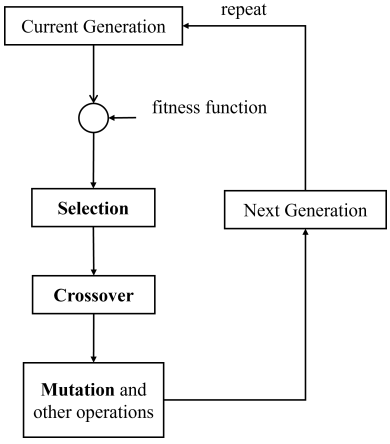


- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*

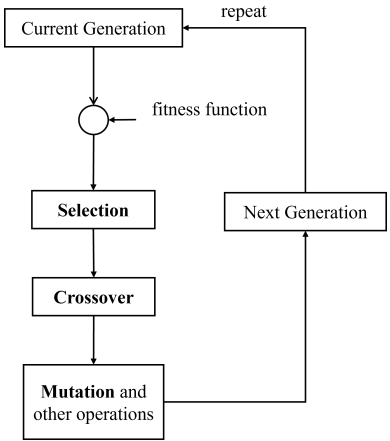
$$fitness(x) = f(x)$$

Genetic Algorithms

2 Background and Related Work

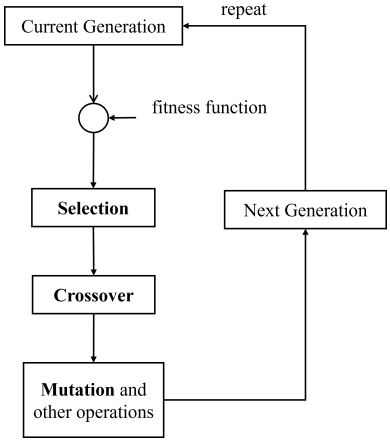


- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic

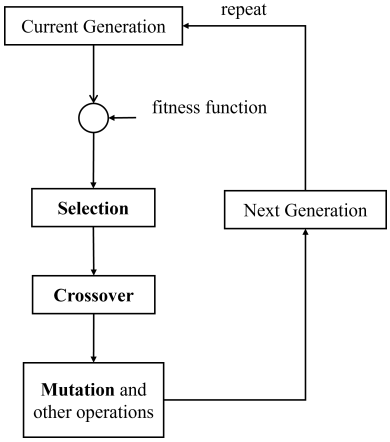
Genetic Algorithms



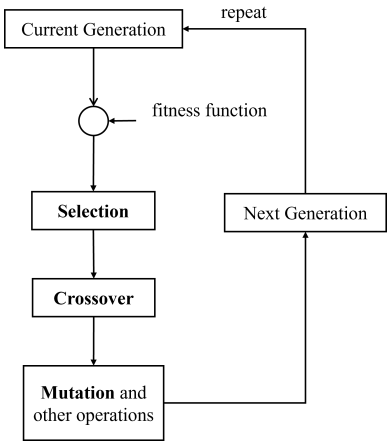
- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic: Elitism
    $\text{argmax}_x fitness(x)$

Genetic Algorithms

2 Background and Related Work



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic

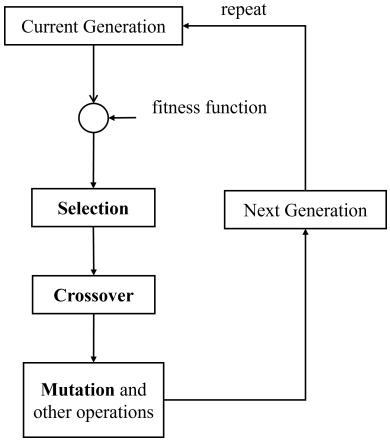


- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic: Roulette wheel selection

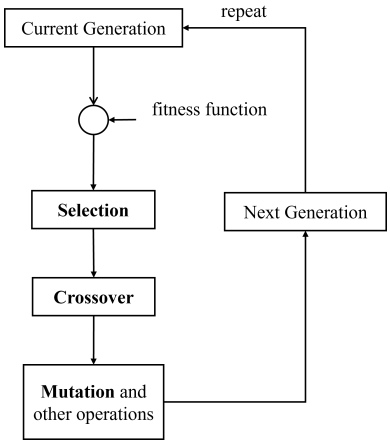
$$P(x) = \frac{fitness(x)}{\sum fitness(x)}$$

Genetic Algorithms

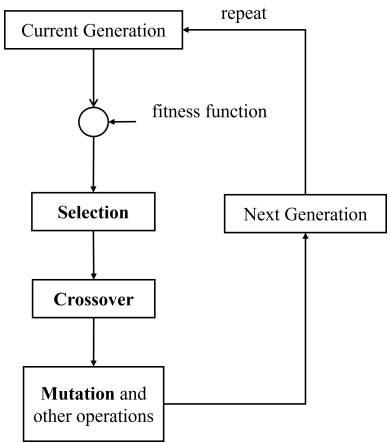
2 Background and Related Work



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic

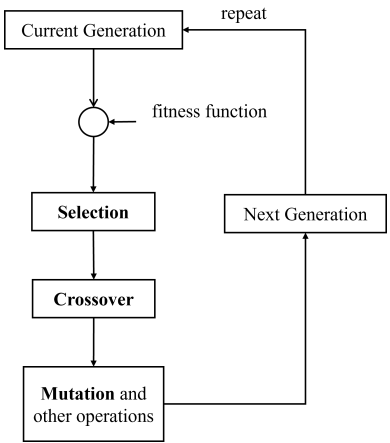


- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes

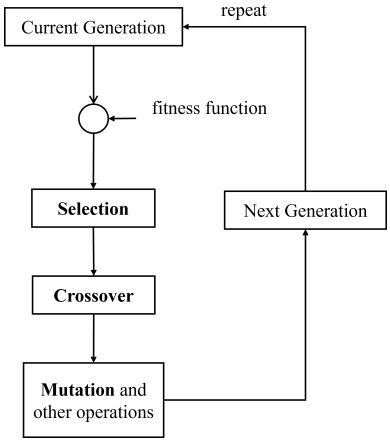
$$Crossover(x_1, x_2) = y$$



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes

  $$Crossover(x_1, x_2) = y$$

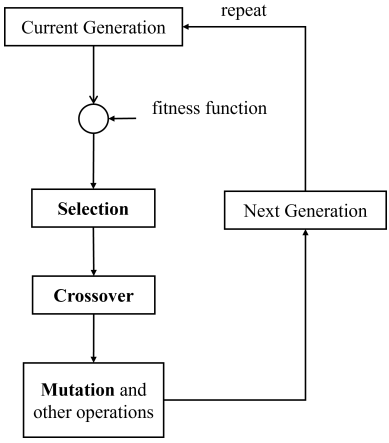
- Execute *mutation* operation with low probability



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes

  $$Crossover(x_1, x_2) = y$$
- Execute *mutation* operation with low probability

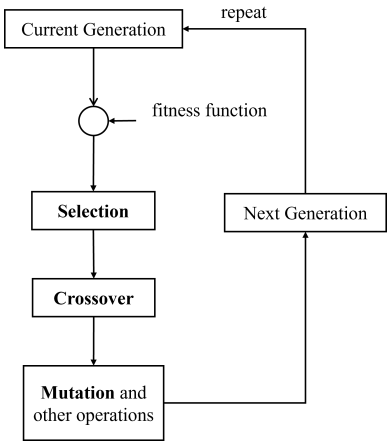
  $$Mutation(y) = y', \text{ with } p_m$$



- First generation is arbitrarily, or $x = random()$
- Evaluate their *fitness*
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes

  $$Crossover(x_1, x_2) = y$$
- Execute *mutation* operation with low probability

  $$Mutation(y) = y', \text{ with } p_m$$
- *Allow the integration of multiple objectives*



- First generation is arbitrarily, or $x = random()$
- ~~Evaluate their *fitness*~~ **a fitness gradient may not always be available**
- *Select* members of the population:
  - Deterministic
  - Stochastic
- Implement *crossover* operation on the reproduced chromosomes

  $$Crossover(x_1, x_2) = y$$
- Execute *mutation* operation with low probability

  $$Mutation(y) = y', \text{ with } p_m$$
- *Allow the integration of multiple objectives*

Novelty Search

- *Exploiting open-endedness to solve problem through the search for novelty*
- Lehman et al., 2008
- Replaces the search for a greater fitness by a search for novel individuals, aim at finding the most novel solutions at each generation
- Increasingly diverse solutions in an open-ended fashion!

Novelty Score

$$\rho(x) = \frac{1}{k} \sum_{k=1}^{k} dist(x, \mu_i)$$

- $\{\mu_1, \mu_2, ..., \mu_k\}$: the set of k-closest individuals to x in archive
- $\rho(x)$: sparseness of x
- $dist$: distance metric

- The set of individuals used to measure the novelty of new solutions corresponds to the current population, plus an archive of previous individuals.

Novelty Search

Novelty Score

$$\rho(x) = \frac{1}{k}\sum_{k=1}^{k} dist(x, \mu_i)$$

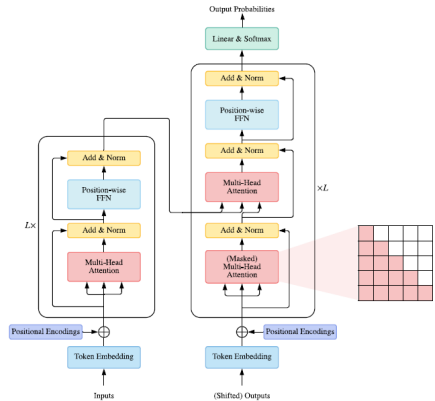
- $\{\mu_1, \mu_2, ..., \mu_k\}$: the set of k-closest individuals to x in archive
- $\rho(x)$: sparseness of x
- $dist$: distance metric

- The set of individuals used to measure the novelty of new solutions corresponds to the current population, plus an archive of previous individuals.
- NS guides generation towards increasingly diverse solutions in an open-ended fashion.

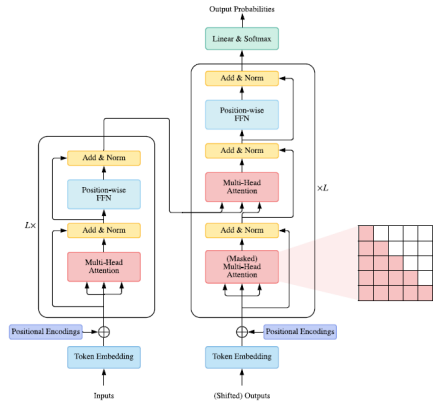
Sequence Modelling and Transformer

- Classic approaches to sequence modelling are RNNs (Rumelhart et al.[4]) and LSTM networks (Hochreiter et al. [3])
  - fading memory
  - limited scalability
- Transformers (Vaswani et al.[5]) can address both challenges, can be parallelized and achieve good scalability!

- ”Attention is all you need” (Vaswani et al., 2017)
- **Encoder**:
- **Decoder**:



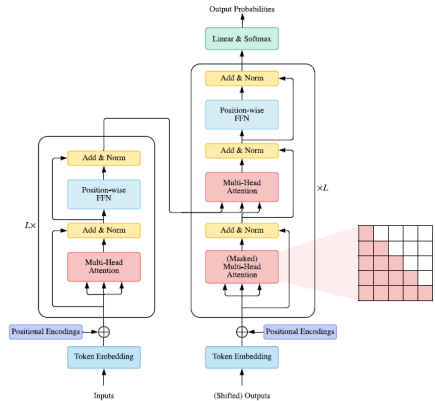
- ”Attention is all you need” (Vaswani et al., 2017)
- **Encoder**:
  - N identical layers (Multi-Head Attention and FFN)
  - Residual connections
- **Decoder**:



Sequence Modelling and Transformers

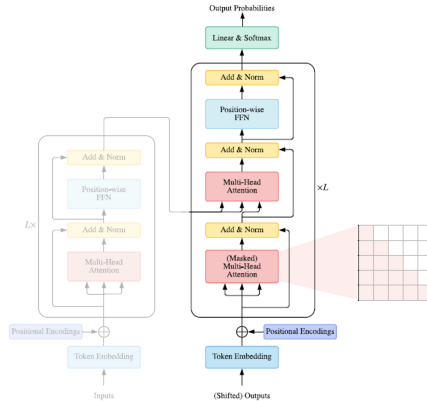
2 Background and Related Work

- ”Attention is all you need” (Vaswani et al., 2017)
- **Encoder**:
  - N identical layers (Multi-Head Attention and FFN)
  - Residual connections
- **Decoder**:
  - N identical layers (Multi-Head Attention, FFN and Masked Multi-Head Attention)
  - Residual connections

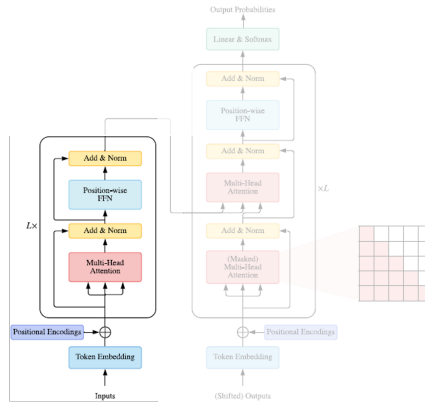


Sequence Modelling and Transformers

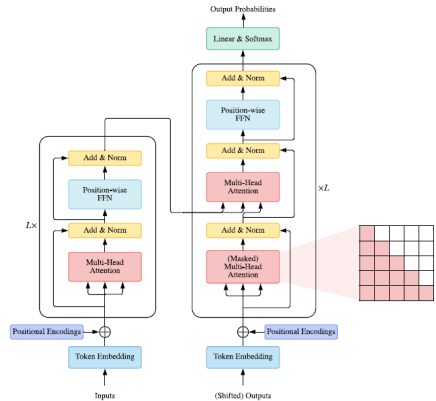
- ”Attention is all you need” (Vaswani et al., 2017)
- **Encoder**:
  - N identical layers (Multi-Head Attention and FFN)
  - Residual connections
- **Decoder**:
  - N identical layers (Multi-Head Attention, FFN and Masked Multi-Head Attention)
  - Residual connections
- *GPT consists of a stack of decoder layers.*



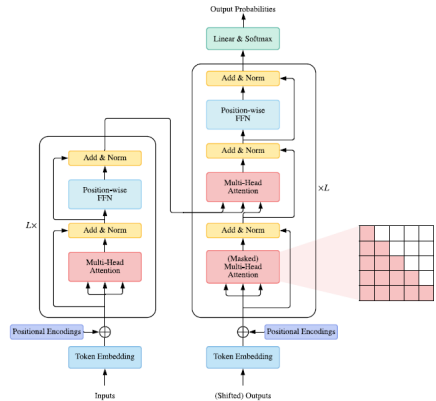
- ”Attention is all you need” (Vaswani et al., 2017)
- **Encoder**:
  - N identical layers (Multi-Head Attention and FFN)
  - Residual connections
- **Decoder**:
  - N identical layers (Multi-Head Attention, FFN and Masked Multi-Head Attention)
  - Residual connections
- *BERT consists of a stack of encoder layers.*



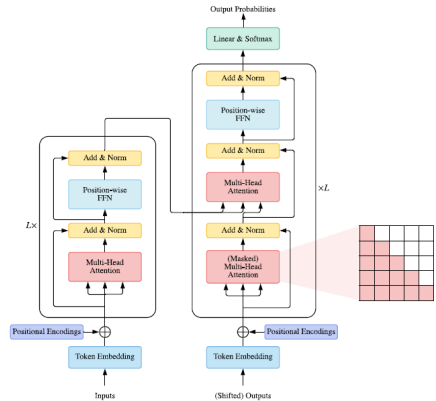
- Enable LLMs to learn from massive datasets



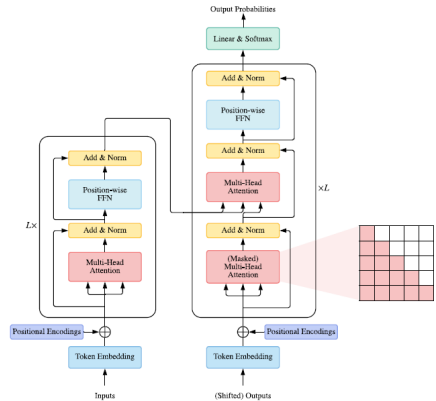
- Enable LLMs to learn from massive datasets
  - Using pre-trained model weights as a weight initialization for new tasks.



- Enable LLMs to learn from massive datasets
  - Using pre-trained model weights as a weight initialization for new tasks.
- *Evolution through Large Models* (Lehman et al., 2022)



- Enable LLMs to learn from massive datasets
  - Using pre-trained model weights as a weight initialization for new tasks.
- *Evolution through Large Models* (Lehman et al., 2022)
  - LLM diff model
  - used as a ”mutation operator”



Sequence Modelling and Transformers

2 Background and Related Work

- Enable LLMs to learn from massive datasets
  - Using pre-trained model weights as a weight initialization for new tasks.
- *Evolution through Large Models* (Lehman et al., 2022)
  - LLM diff model
  - used as a ”mutation operator”
- Produce incredibly diverse mutations, vary increasingly over the course of the GA!

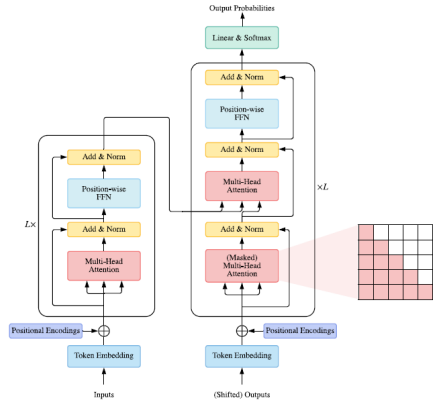


Table of Contents

3 Open-Ended Level Generation through LLMs

- ▶ Introduction
- ▶ Background and Related Work
- ▶ Open-Ended Level Generation through LLMs
- ▶ Experiments and Results
- ▶ Conclusion

Level Representation

- Use the levels provided in the Video Game Level Corpus (Summerville et al., 2016)

Level Representation

- Use the levels provided in the Video Game Level Corpus (Summerville et al., 2016)
- Utilize a set of path-annotated levels, taken from Super Mario Bros. and Super Mario Bros.: The Lost Levels (in total 37 levels)

Level Representation

- Use the levels provided in the Video Game Level Corpus (Summerville et al., 2016)
- Utilize a set of path-annotated levels, taken from Super Mario Bros. and Super Mario Bros.: The Lost Levels (in total 37 levels)
  - Stitch selected levels together, make one giant level!

Level Representation

- Use the levels provided in the Video Game Level Corpus (Summerville et al., 2016)
- Utilize a set of path-annotated levels, taken from Super Mario Bros. and Super Mario Bros.: The Lost Levels (in total 37 levels)
  - Stitch selected levels together, make one giant level!
- Tokenize string representation into discrete values using Byte Pair Encoding.

Level Representation

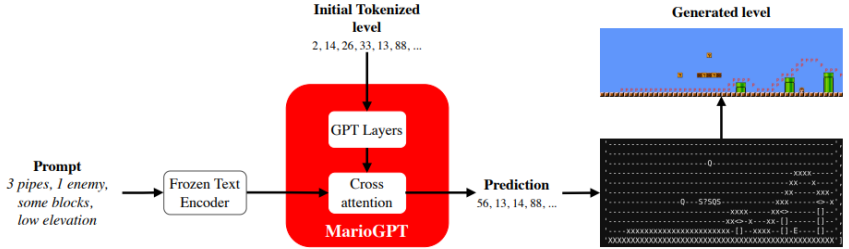
3 Open-Ended Level Generation through LLMs

Tile Type	Symbol	Visualization
Empty	-	
Unbreakable	X	
Breakable	S	
Question Block	? / Q	
Coin	o	
Enemy	E	
Left pipe top	<	
Right pipe top	>	
Left pipe lower	[	
Right pipe lower	]	
Cannon Top	B	
Cannon Body	b	
Path	x	

Unique Mario tiles (Sudhakaran et al., 2023)

MarioGPT Model

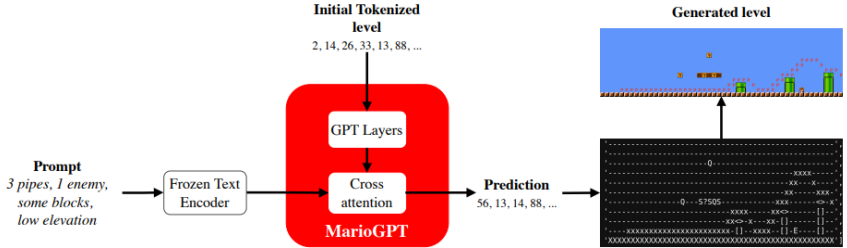
3 Open-Ended Level Generation through LLMs



- **DistilGPT2** (Li et al., 2021) is MarioGPT's backbone.

MarioGPT Model

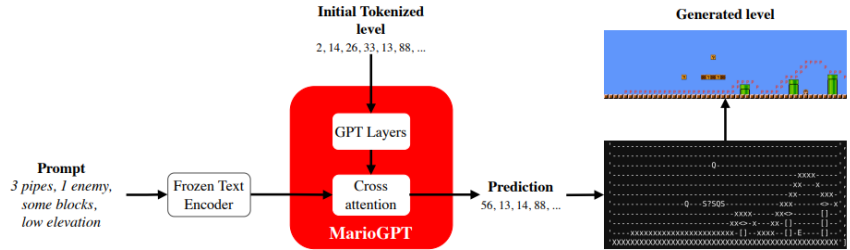
3 Open-Ended Level Generation through LLMs



- **DistilGPT2** (Li et al., 2021) is MarioGPT's backbone.
- **Cross attention** layer takes prompt information into account.

MarioGPT Model

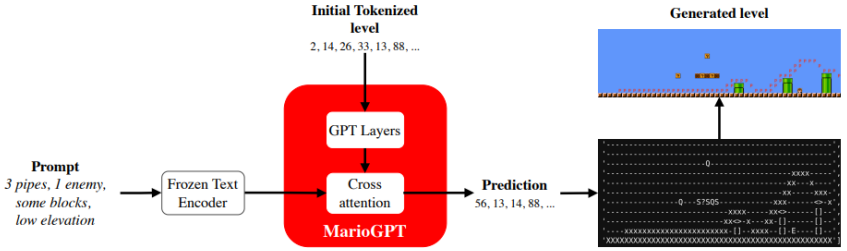
3 Open-Ended Level Generation through LLMs



- **DistilGPT2** (Li et al., 2021) is MarioGPT's backbone.
- **Cross attention** layer takes prompt information into account.
- 96M parameters of DistilGPT2 (86M) and Cross attention weights (10M).

MarioGPT Model

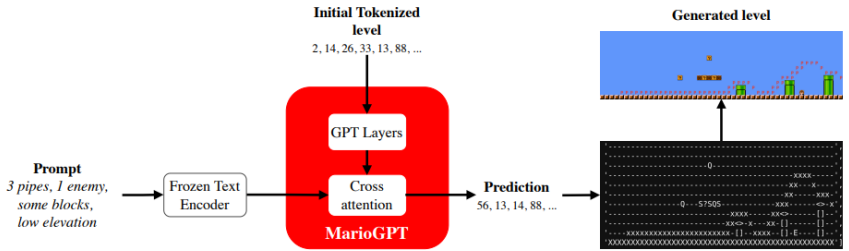
3 Open-Ended Level Generation through LLMs



- Prompts are encoded through BART (Lewis et al., 2019).

MarioGPT Model

3 Open-Ended Level Generation through LLMs



- Prompts are encoded through BART (Lewis et al., 2019).
- Prompt's feature vector is combined with the actual level sequence in the cross attention layers.

MarioGPT Model

3 Open-Ended Level Generation through LLMs

- Prompts:
  - { *no*, *little*, *some*, *many*, *[0-1000]* } *pipes*
  - { *no*, *little*, *some*, *many*, *[0-1000]* } *enemies*
  - { *little*, *some*, *many*, *[0-1000]* } *blocks*
  - { *low*, *high* } *elevation*

MarioGPT Model

3 Open-Ended Level Generation through LLMs

- Prompts:
  - { *no*, *little*, *some*, *many*, *[0-1000]* } *pipes*
  - { *no*, *little*, *some*, *many*, *[0-1000]* } *enemies*
  - { *little*, *some*, *many*, *[0-1000]* } *blocks*
  - { *low*, *high* } *elevation*
- Example:
  - *"no pipes, many enemies, low elevation"*
  - *"many pipes, many enemies, many blocks"*

MarioGPT Model

3 Open-Ended Level Generation through LLMs

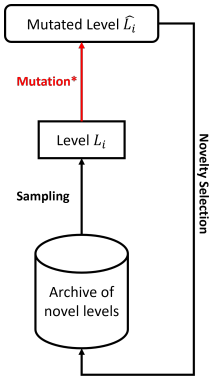
tile	no	little	some	many
pipes	0	1	2	5
enemies	0	1	3	7
blocks	0	50	75	176

Prompt Quantiles and corresponding counts within a 50 column window

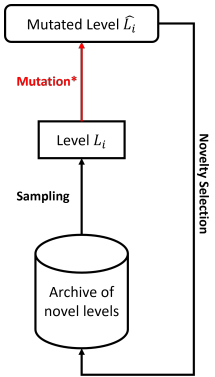
Open-Ended Mario Level Generation with Novelty Search

3 Open-Ended Level Generation through LLMs

- Not only generate levels with diverse physical features, but also elicit a wide range of **player behavior**.
  - a challenge for many algorithms
  - the use of an external agent for evaluation
- **NS-MarioGPT**
  - integrate MarioGPT within a novelty search augmented genetic algorithm

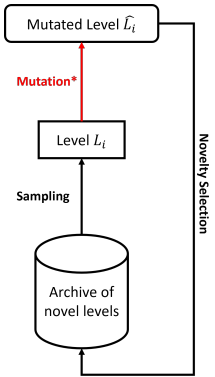


- Initialize archive with 30 levels.



- Initialize archive with 30 levels.
- Novelty score: k-means

$$\rho(x) = \frac{1}{k}\sum_{k=1}^{k} dist(\varphi_{\mathcal{B}}(x), \varphi_{\mathcal{B}}(\mu_i))$$



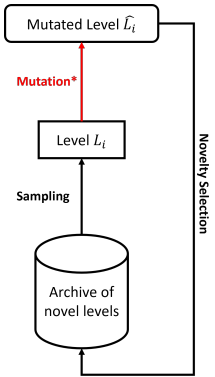
- Initialize archive with 30 levels.
- Novelty score: k-means

$$\rho(x) = \frac{1}{k}\sum_{k=1}^{k} dist(\varphi_{\mathcal{B}}(x), \varphi_{\mathcal{B}}(\mu_i))$$

- Use **predicted** player paths as basis for $\varphi_{\mathcal{B}}$.

Open-Ended Mario Level Generation with Novelty Search

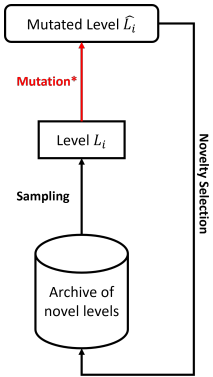
3 Open-Ended Level Generation through LLMs



- Initialize archive with 30 levels.
- Novelty score: k-means

$$\rho(x) = \frac{1}{k}\sum_{k=1}^{k} dist(\varphi_{\mathcal{B}}(x), \varphi_{\mathcal{B}}(\mu_i))$$

- Use **predicted** player paths as basis for $\varphi_{\mathcal{B}}$.
- Archive only stores levels with an acceptable novelty score ($\rho(x) > threshold$).



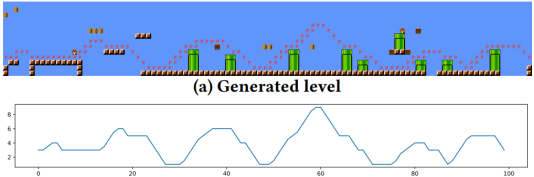
- Initialize archive with 30 levels.
- Novelty score: k-means

$$\rho(x) = \frac{1}{k}\sum_{k=1}^{k} dist(\varphi_{\mathcal{B}}(x), \varphi_{\mathcal{B}}(\mu_i))$$

- Use **predicted** player paths as basis for $\varphi_{\mathcal{B}}$.
- Archive only stores levels with an acceptable novelty score ($\rho(x) > threshold$).
- In this paper, $k = 4$.

Open-Ended Mario Level Generation with Novelty Search

3 Open-Ended Level Generation through LLMs



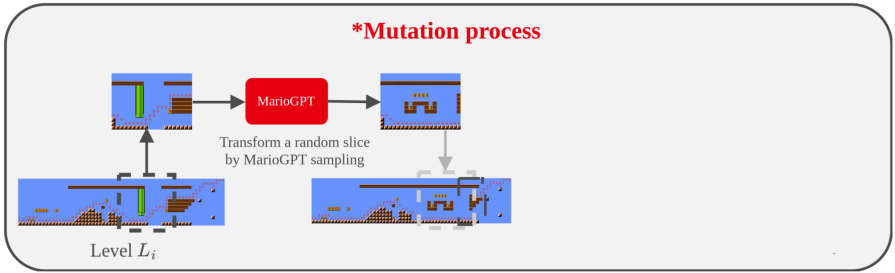
(a) Generated level

(b) Extracted path from level

(c) Behavior characteristic of level: smoothed path using a moving average with a window of 5 coordinates

Open-Ended Mario Level Generation with Novelty Search

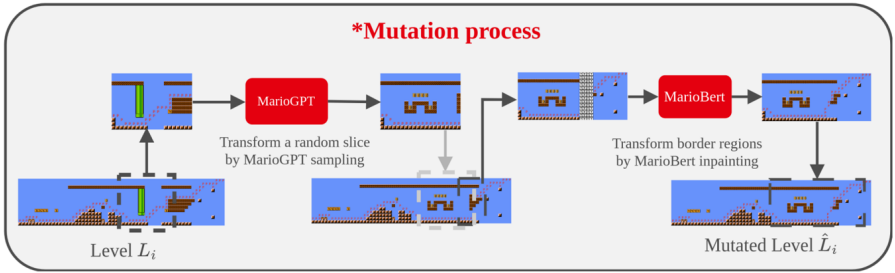
3 Open-Ended Level Generation through LLMs



- Model transforms a *randomly* picked slice (40-80 columns) of a level with **MarioGPT**.

Open-Ended Mario Level Generation with Novelty Search

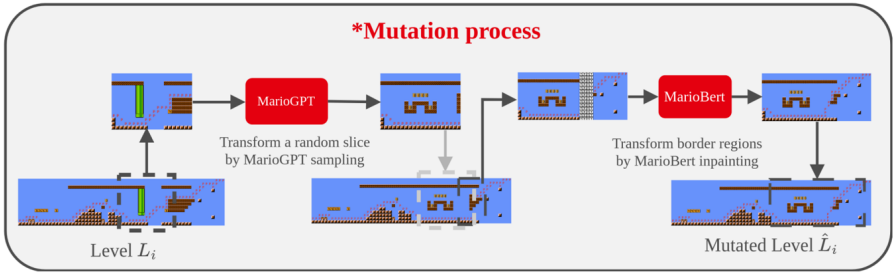
3 Open-Ended Level Generation through LLMs



- Model transforms a *randomly* picked slice (40-80 columns) of a level with **MarioGPT**.
- To further improve path consistency, **MarioBert** (fine-tuned mask prediction model) is used to inpaint its border region.

Open-Ended Mario Level Generation with Novelty Search

3 Open-Ended Level Generation through LLMs



- Model transforms a *randomly* picked slice (40-80 columns) of a level with **MarioGPT**.
- To further improve path consistency, **MarioBert** (fine-tuned mask prediction model) is used to inpaint its border region.
- Smoothly join the mutated slice and the rest of level.

Table of Contents

4 Experiments and Results

- ▶ Introduction
- ▶ Background and Related Work
- ▶ Open-Ended Level Generation through LLMs
- ▶ Experiments and Results
- ▶ Conclusion

Reconstruction and Sampling Quality

Its proved in original paper that MarioGPT (using a pretrained GPT2 model) outperforms other baselines with regards to tile prediction.

Model	Tile Acc.	Path Acc.	Promptable?
LSTM	46%	39%	NO
from-scratchMarioGPT	31%	23%	YES
adapterMarioGPT	21%	11%	YES
MarioGPT	93%	91%	YES

Table: Training Reconstruction Accuracy – Validation Set

Reconstruction and Sampling Quality

4 Experiments and Results

To assess MarioGPT's capability in generating diverse and playable levels, we focus on two questions :

1. **Playability of levels generated by MarioGPT** : Are MarioGPT's generated levels solveable?
2. **Learning capability of MarioGPT**: Is MarioGPT memorizing?

Reconstruction and Sampling Quality

4 Experiments and Results

(1) Measuring Playability of Levels

- To test for playability, we deploy Robin Baumgarten's A* agent in generated levels, each level was played for 5 times.
  - Playable levels : 201
  - Not playable levels : 39

83.75% generated levels can be solved(it is 88.33% in original paper.)

Reconstruction and Sampling Quality

4 Experiments and Results

(1) Measuring Playability of Levels

- To test for playability, we deploy Robin Baumgarten's A* agent in 240 generated levels, each level was played for 5 times.
  - Playable levels : 201
  - Not playable levels : 39

83.75% generated levels can be solved(it is 88.33% in original paper.)

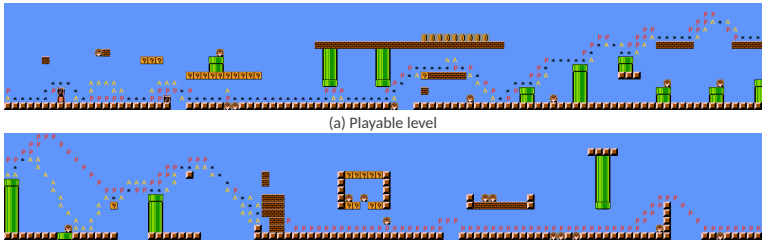
Playable	Not Playable	All
1.8009	8.68538	2.4375

Table: MAE between paths suggested by model and Baumgarten's A* agent **in original paper**.

Playable	Not Playable	All
1.14	4.55	1.55

Table: MAE between paths suggested by model and Baumgarten's A* agent **in our experiment**.

(1) Measuring Playability of Levels



(a) Playable level

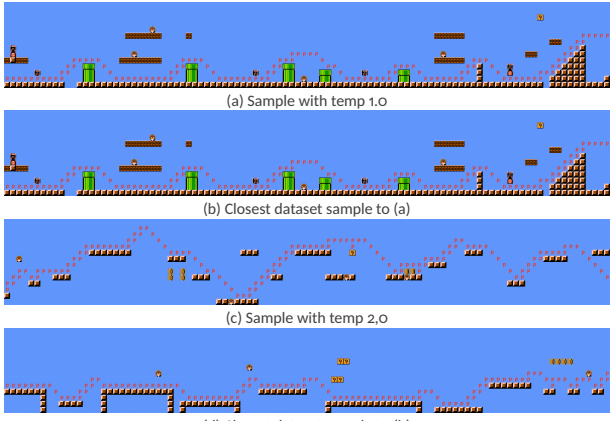
(b) Non-playable level

Figure: A* vs. MarioGPT generated paths. A* agent (denoted as A), and model suggestion (denoted as P) Positions where both trajectories overlap are marked with *.

(2) Is MarioGPT memorizing?

- LLMs are incredibly powerful, but sometimes overfit extremely and end up regurgitating training data.
- Add some randomness in predictions in the form a tunable *temperature* parameter.

(2) Is MarioGPT memorizing?



(a) Sample with temp 1.0

(b) Closest dataset sample to (a)

(c) Sample with temp 2.0

(d) Closest dataset sample to (b)

Figure: Generated levels vs closest in dataset.

Guided Level Generation via Prompting

- Evaluate the prompting ability of MarioGPT by comparing generated levels with the prompt descriptions.

pipes	enemies	blocks	elevation
75%	66.25%	81.25%	76.25%

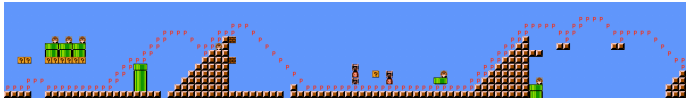
Table: Prompt vs actual description accuracy **in our experiments**.

pipes	enemies	blocks	elevation
81%	68%	92%	76%

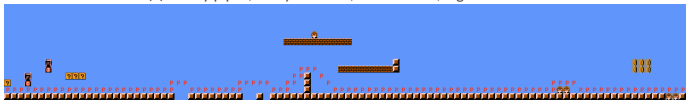
Table: Prompt vs actual description accuracy **in original paper**.

Guided Level Generation via Prompting

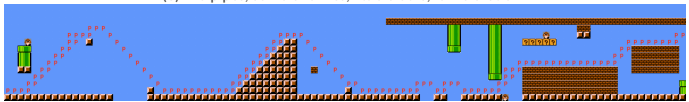
4 Experiments and Results



(a) "many pipes, many enemies, some blocks, high elevation"



(b) "no pipes, some enemies, little blocks, low elevation"



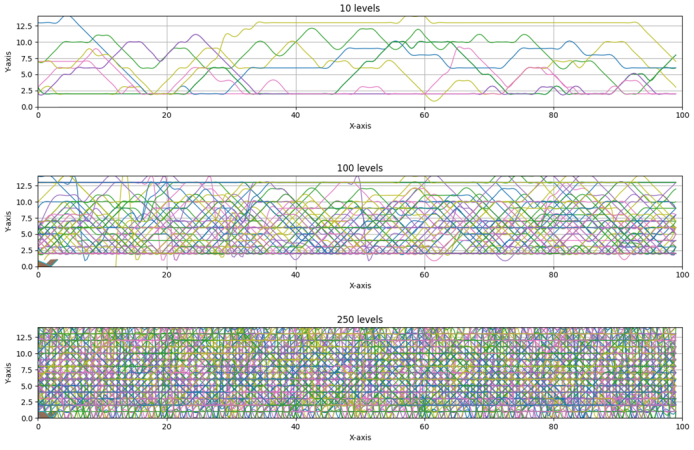
(c) "some pipes, some enemies"

(d) Prompt that does not exist in the dataset: "many pipes, no enemies, many blocks"

Figure : Multiple prompt generations from a single seed block.

Generating Diverse Levels with Novelty Search

4 Experiments and Results

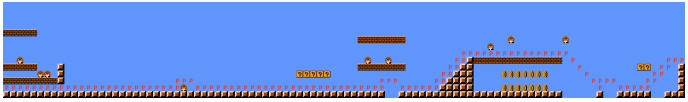


Archive after 10, 100, 250 levels.

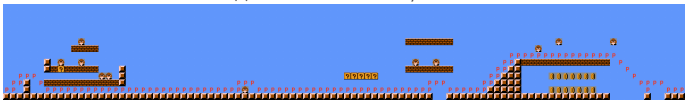
Figure: Generated path populations during novelty search. Each line is a path through a level.

Generating Diverse Levels with Novelty Search

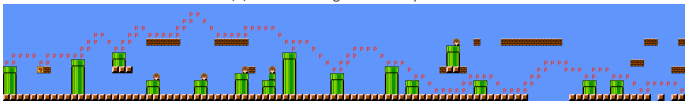
4 Experiments and Results



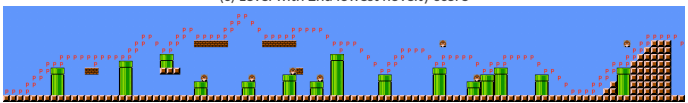
(a) Level with lowest novelty score



(b) Level with highest novelty score



(c) Level with 2nd lowest novelty score



(d) Level with 2nd highest novelty score

Figure: Comparison of most and least novel levels in the archive.

Table of Contents

5 Conclusion

- ▶ Introduction
- ▶ Background and Related Work
- ▶ Open-Ended Level Generation through LLMs
- ▶ Experiments and Results
- ▶ Conclusion

Conclusion

5 Conclusion

- MarioGPT is a fine-tuned LLM based on GPT2, capable of generating diverse and guided levels through language prompts.
- MarioGPT is able to :
  - predict player interaction in generated levels.
  - generate diverse and playable environments.
  - reduce the need for expensive. external agent interactions (MarioGPT can generate playable levels approximately 82% of the time)
- MarioGPT can generate open-ended and functional content with a diversity-driven algorithm (novelty search).

Future Work

- Leverage the scalability of LLMs and train MarioGPT on bigger, more detailed annotated levels.
- Incorporating human feedback into the level generation process through reinforcement learning from human feedback (RLHF).
- Opens the door to more controllable and diverse PCG systems.