Bridging Natural Language and Code: Transforming Free-Form Sentences into a Sequence of Unambiguous Sentences with LLM.

Nikita Kiran Yeole, Michael S. Hsiao

Department of Computer Science. Virginia Tech, USA
Email address: *nikitay@vt.edu*

VIRGINIA TECH

IARIA

Myself

Nikita Yeole received the bachelor's degree in Computer Science(CS) from SNDT University, India. She is currently Pursuing Masters in CS from Virginia Polytechnic and State University, USA.

Her research interest lies in Artificial Intelligence, particularly interested in Natural language processing and Natural Language Programming.

Introduction about Natural Language Programming

- Natural Language Programming :

➢ Attempts to convert instructions written in free-form natural language into executable program code.

➢ Envisions a world in which everyone can program machines without understanding the intricacies of conventional programming languages.

Code generation from Natural Language using Gen-AI

While generative AI has shown some success in producing code from natural language text, the code that is produced may not adhere to the intent of the input text.

Now, The user can do one of two things:

(1) manually modify the generated code
(2) re-write the natural language text and try to generate new code.

Educational Use

- Natural language is increasingly applied in education
  - personalized AI tutoring
  - interactive learning, etc
- Becomes important for a user to learn to write unambiguous input text
  - a necessary skill behind the thought processes in coding.
- The ability to instruct a machine in natural language
  - bridges the gap between human thought processes and the digital world
  - Makes technology more accessible and intuitive for students.

What is the problem we are trying solve?

- Many factors associated with Natural language Instructions which makes NLPg very Challenging.
- We address four main challenges of Natural Language.

1. **The ambiguity** in the sentences.
2. **High level verbose** and **Descriptive** sentences.
3. **Complex and Compound** Sentences.
4. **Invalid** or **erroneous** sentences given by human.

Example 1: Ambiguity:

“When the rabbit touches a rock, it explodes.”

Here, the phrase containing the pronoun ’it’ creates uncertainty in this sentence.

According to one view, the rabbit explodes after touching the rock, whereas the other contends that the rock explodes.

Example 2: High level verbose and Descriptive sentences.

“In a mysterious realm, a lone pointer and some aliens engage in a cosmic dance. When the pointer touches an alien, it changes colors: original to purple, purple to pink. Pink aliens explodes.”

Here, The sentences provided are verbose with extraneous words and phrases.

Example 3 : Complex and compound sentence:

"When the carrot turns into a diamond before the carrot touches a fox, the score increases."

Example 4: Invalid/erroneous Sentences:

"Brick spawns at the bottom. 14 cheese at the top in rows. Ball in the middle. w is up. s is down. brick touches border bounce. ball touches cheese bounces back."

Solution

Build an intermediate system which:

- Transforms the Free form sentences into sequences of simple sentences with a clear subject, verb, and object structure using LLM.
- Promotes a paradigm where instead of the user conforming to the machine, the machine adapts to grasp the user's intent
- Bridges the Natural Language and the code.

Why LLM

- Huge strides have been made in recent years.
- Handles nuances in NL.
- Trained on huge data.
- Useful for Context analysis.

Proposed Design

- The proposed approach contains multiple components in the following steps:

(1) Question Answering,

(2) Sentence Reframing,

(3) Sentence Decomposition

- Additional approaches, such as few-shot learning and model fine-tuning, revealed that the above three steps performed better overall.

Input: Free form Game description

Example Question

What are the default actions performed by the object mentioned in the passage? Write in format of {object1 : {default action}}.

Question Answering

Example Rule

Default Actions : Write every default action if exists in the format "{object1 } {verb of action}{object 2}" for every character separately.

Sentence reframing

Example Template

Input Template: *When [Entity1] {action} [Entity2] before [Entity3] {action} [Entity4], [Outcome].*
Output Template: *When [Entity1] {action} [Entity2], [Entity1] becomes [Attribute1]. When [Entity3] {action} [Entity4], [Entity3] becomes [Attribute2]. When [Entity1] is [Attribute1] and [Entity3] is not [Attribute2], [Outcome].*

Sentence decomposition

Output: Transformed Sentence

Functional Game

Question Answering

- We extract vital details

➢ Entities like: *the objects, location of the objects, default actions, interactions between the objects, the keyboard or mouse controls.*

- recognize specific conditions that might trigger

➢ certain actions : *such as color changes or score increments*

- **Predetermined format :** explicitly stated in the question.
- **Predefined lists** are a key component of our QA system. The permissible objects, verbs, and designated formats for the antecedent and consequent responses are all included in these lists.

Examples questions from the Prompt

The question we used to extract the details are listed below:

1. Write the name of all objects and number of objects mentioned in the passage. If the number of object is not mentioned , assume it as 10.
2. The default actions are the actions which are performed without any required condition to occur. What are the default actions performed by the object mentioned in the passage? Write in format of object1 : default action.

Example

Input Sentence: The rabbit wanders, reversing at borders. The fox wanders, chasing the rabbit when spotting the rabbit.

Response:
Objects mentioned in the passage: **rabbit (1), fox (1)**
Default actions:- **rabbit: wander, fox: wander**
Location of the objects mentioned in the user input passage: **rabbit: Randomly placed fox: Randomly placed**
Border interactions mentioned in the user input passage:
If the rabbit reaches the left border, it reverses.
If the rabbit reaches the right border, it reverses.
If the rabbit reaches the top border, it reverses.
If the rabbit reaches the bottom border, it reverses
Other interactions mentioned in the user input passage: **If the fox spots the rabbit, it chases the rabbit.**

Sentence Reframing

- Main goal :
  - ➢ to use a specified set of rules
  - ➢ to reconstruct the sentences in their basic form
  - ➢ in the format subject-verb-object.

- Example rule from the Prompt:

  Default Actions : Write every default action if exists in the format "object1 verb of action object 2" for every character separately.

  Response: **The rabbit wanders. The fox wanders.**

Sentence Decomposition

Complicated sentence structures frequently result in misunderstandings, especially those that illustrate complex connections between elements.

Example: *"When the rabbit touches the fox, the fox turns into a carrot."*

This stage:

- standardizes the complex interactions between objects
- and presents them in a structured manner.

Example Format from the prompt

Input Template: "When [Object A] [action with] [Object B], [Object B] [turns into another object].”

Generalized Output: "When [Object A] [action with] [Object B], [Object B] becomes mutated. When [Object B] is mutated, it turns into [Object C].”

Response: ***”When the rabbit touches the fox, the fox becomes mutated. When the fox is mutated, it turns into a carrot."***

Example Format from the Prompt

Input Template: When [Entity1] action [Entity2] before [Entity3] action [Entity4], [Outcome].
Generalized Output: When [Entity1] action [Entity2], [Entity1] becomes [Emotion1]. When [Entity3] action [Entity4], [Entity3] becomes [Emotion2]. When [Entity1] is [Emotion1] and [Entity3] is not [Emotion2], [Outcome].

User Input: *When the fox touches the carrot before the rabbit, the fox dies.*

Response: ***When a fox touches a carrot, the fox becomes happy. When a rabbit does not touch the carrot, the rabbit becomes not happy. When the fox is happy and the rabbit is not happy, the fox dies.***

When an apricot touches a border, the speed of the apricot is -1. When rabbit touches a apricot, rabbit turns into diamond.

Objects (apricots, rabbits, borders, diamonds), the default actions (apricots and rabbits move), and the conditional actions (speed decrease for apricots, transformation for rabbits)

Re-framed sentence: If the apricot touches a border, the speed of the apricot decreases. If the rabbit touches a carrot, the rabbit turns into a diamond.

Decomposed and Reconstructed: If the apricot touches a border, the speed of the apricot decreases. When the rabbit touches a carrot, the rabbit becomes mutated. When the rabbit is mutated, it turns into a diamond.

Evaluation Methodology

1. Data Collection :

- 800 free write Sentences (Game descriptions) which are identified as potentially problematic.
- 200 free write Sentences : Identified accurate.
- Used as the LLM's main input.

2. Model Selection:

- GPT-3.5 Turbo
- Made this choice after carefully comparing the performance of GPT-3.5 Turbo and GPT-4.

3. Model Configuration:

- Temperature set to 0 : to guarantee deterministic performance from the model effectively eliminating randomness.
- The top_p parameter was set to 1: implies that at each stage of the generation process, the model will only take into account the tokens that are the most likely.

4. Input to the Model

- The user prompt constitutes the primary interaction point with the user.
- The system prompt serves as a tool to direct the model towards a specific context or mode of operation.

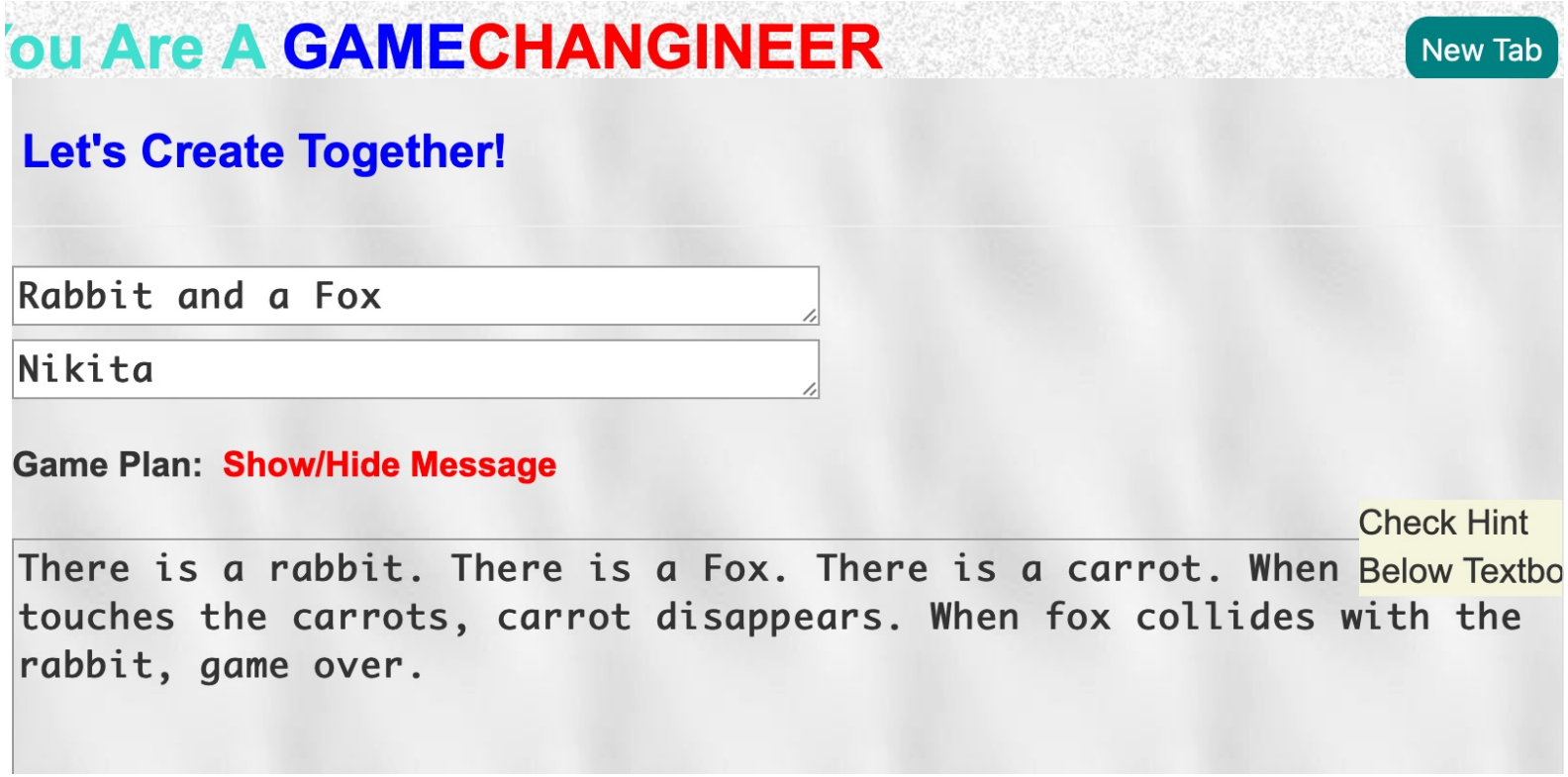
5. Accuracy and Assesment

Integration with GameChangineer Platform

- Educational Platform

- To Practice logical reasoning, problem solving, algorithmic design, critical and computational thinking.

- Develops a functional game using simple English sentences.



Powered by Google Translate

"Rabbit and a Fox" by Nikita

Quick check of your gameplan...

Jump to bottom

Processing your object declarations...

- There is a rabbit -- Understood with 99% certainty.
- There is a Fox -- Understood with 99% certainty.
- There is a carrot -- Understood with 99% certainty.

Post-processing object declarations...
NOTE: the carrot is placed randomly initially.
NOTE: the fox is placed randomly initially.
NOTE: the rabbit is placed randomly initially.
NOTE: Use the map textbox to position randomly placed objects at more exact positions.
NOTE: There is no character controlled by the player.

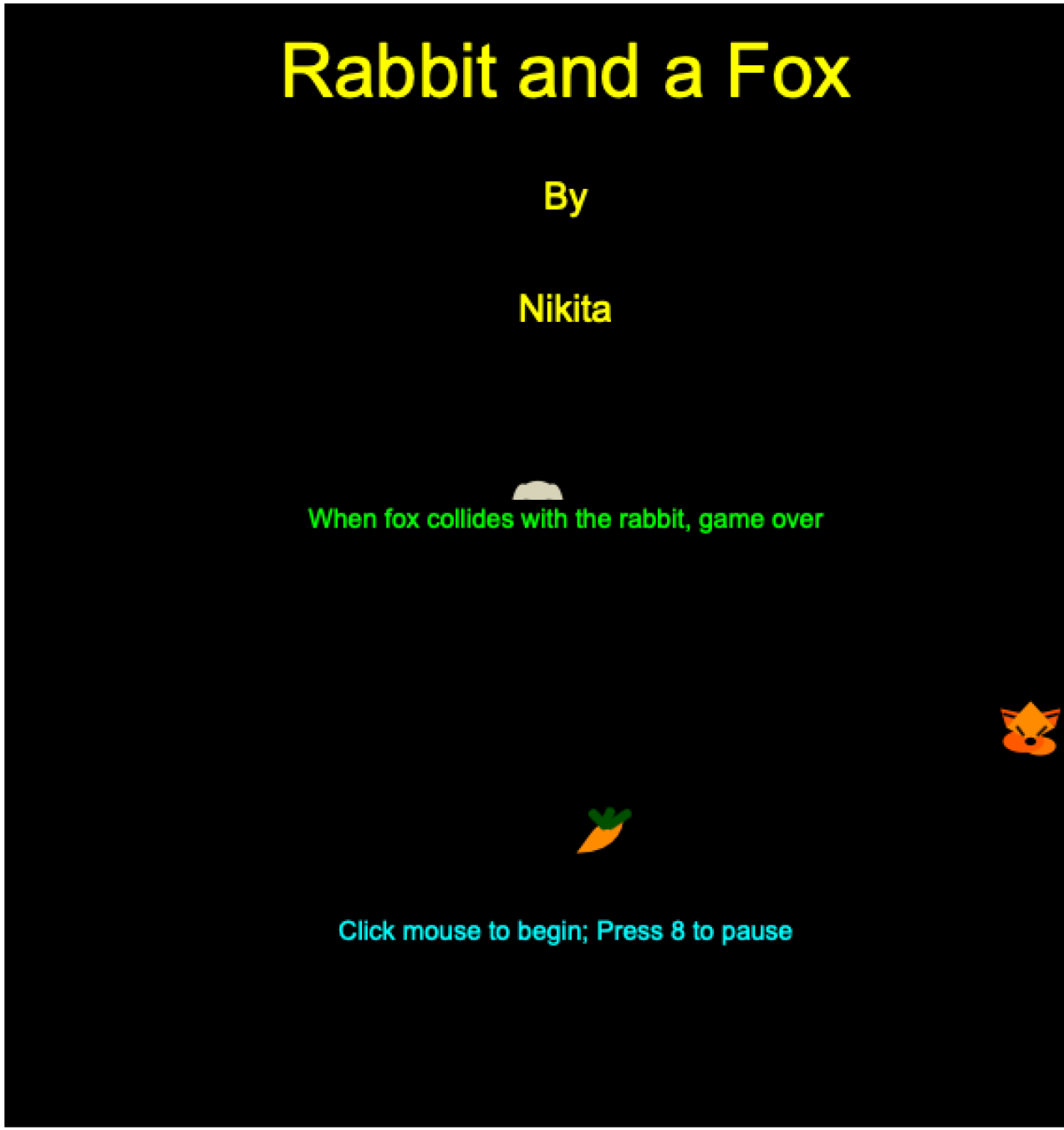
Pre-processing your actions...

- When rabbit touches the carrots, carrot disappears -- 🔊 Listen sen #4 - Locate
  Potential GRAMMATICAL CHECK: are you missing a 'the' between the words 'carrots and carrot'?
  🔊 Listen to the above message

Processing your actions...

- When rabbit touches the carrots, the carrot disappears.-- Understood with 99% certainty. -- note: the object in the consequent will die, disappear, or explode in the cycle/frame immediately after the antecedent event.
- When fox collides with the rabbit, game over.-- Understood with 99% certainty.

Converting your game plan to code...
..... synthesizing code recommendations



Evaluation and results

- Input to the system : User input free form Sentences
- Output of the system : LLM’s Final response

For Evaluation:

- Output was passed to the GameChangineer platform to predict its accuracy.
- Output was also checked manually if the output reflects the intent of the user.
- 90% or higher accuracy on the GameChangineer platform was considered as a successful transformation.

Successful Examples

1. The kitten must touches all the cakes to win without having the fox touch the kitten.

Response: ***When the kitten touches all the cakes, the player wins the game. If the fox touches the kitten, the player loses the game.***

2. fox explodes rabbit when it is the rabbit is touched by it.

Response: ***If the rabbit is touched by the fox, the fox explodes.***

Unsuccessful Examples

1. When the rabbit is yellow, the rabbit touches the fox, the fox explodes.

Response: ***If the rabbit is yellow, the rabbit touches the fox. If the rabbit touches the fox, the fox explodes.***

2. When a ball sees the rock, the ball flees from the rock.

Response: ***When the ball sees the rock, the ball becomes scared and flees from the rock.***

Evaluation and results

TABLE I. Sentence Categorization Results

Category	**Sentence proportion**	**Success Rate**
Grammar/Typing mistakes	22%	75.6%
Ambiguous Sentences	18%	57.7%
Unrealizable Actions	11.6%	50%
Overly Complex/Descriptive	28.4%	59%
Non-problematic sentences	20%	94%
Total Success Rate	100%	68.4%

Findings

- Realistic relations

Input: random rocks fall the top border

Output: When the rocks fall, they hit the ground. When the rocks hit the ground, they stop.

Limitations

- LLMs does not consider every single point in the prompt while giving outputs so it sometimes miss a few instructions given to it.
- Hallucination
- This makes some transformation unsuccessful.

Conclusion

- Successfully transformed **62% of the 800 problematic sentences** into a sequence of unambiguous , simplified sentences that may be efficiently translated into machine-executable code.
- LLMs has shown promise in addressing the inherent difficulties brought about by **verbosity, ambiguities, complexity, and possible error**

Thank you