GDC | March 20-24, 2023
San Francisco, CA

Developing Adventure Game with Free Text Input using NLP

Yusuke Mori
AI Researcher
AI Division
SQUARE ENIX CO., LTD.
moriyus@square-enix.com

#GDC23

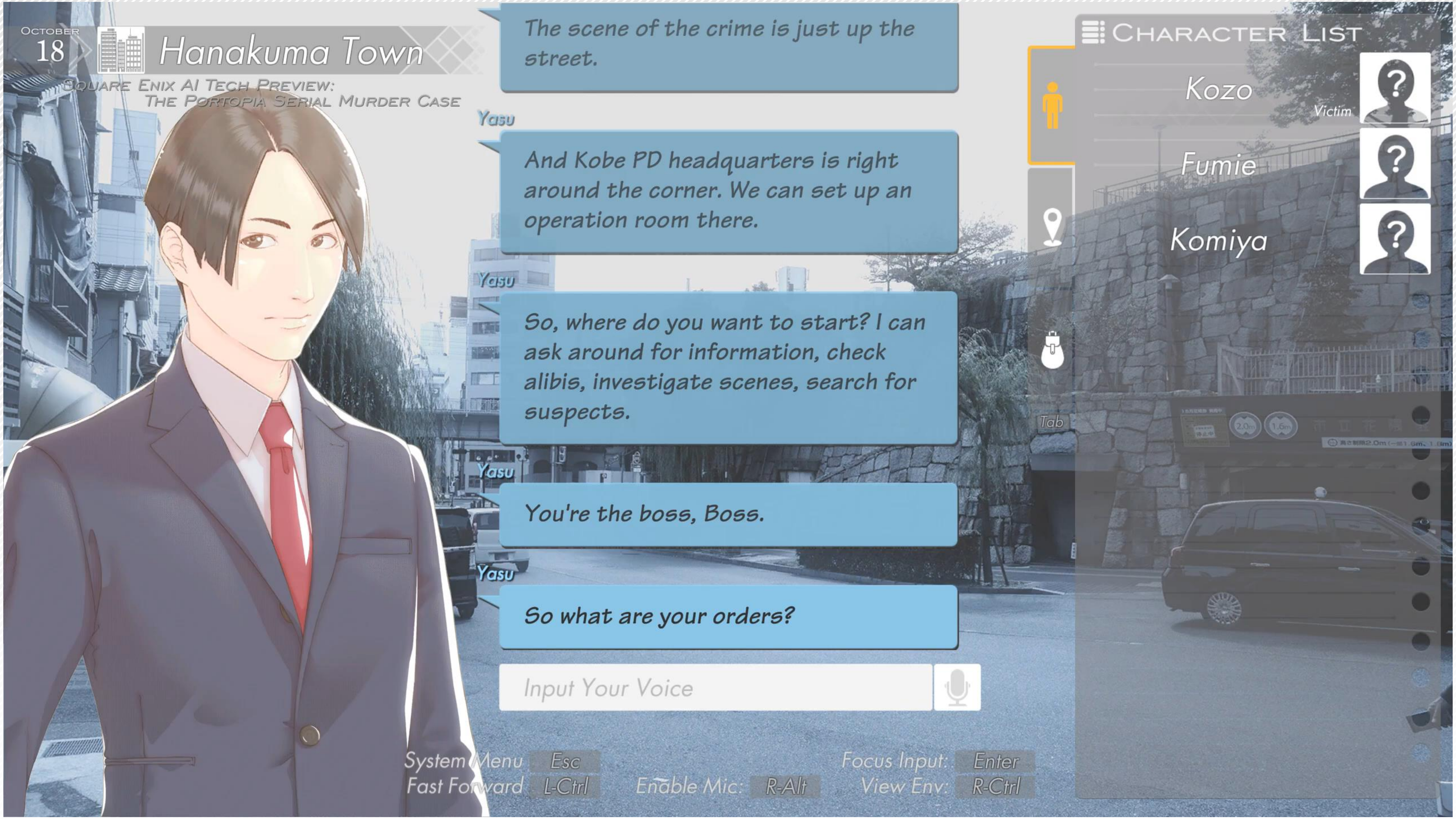
Self-introduction

Yusuke Mori

- Ph.D. in the field of Information Science and Technology
  - From the University of Tokyo, Japan, in 2021
  - Research Keywords:
    - Natural Language Processing (NLP)
    - Creative Writing Support System
    - Reader Emotions
- AI Researcher
  - AI Division, SQUARE ENIX CO., LTD.
  - **Working on the application of NLP to Game AI**

NLP Adventure

Adventure Game with Natural Language Processing AI



The scene of the crime is just up the street.

Yasu

And Kobe PD headquarters is right around the corner. We can set up an operation room there.

Yasu

So, where do you want to start? I can ask around for information, check alibis, investigate scenes, search for suspects.

Yasu

You're the boss, Boss.

Yasu

So what are your orders?

Experience

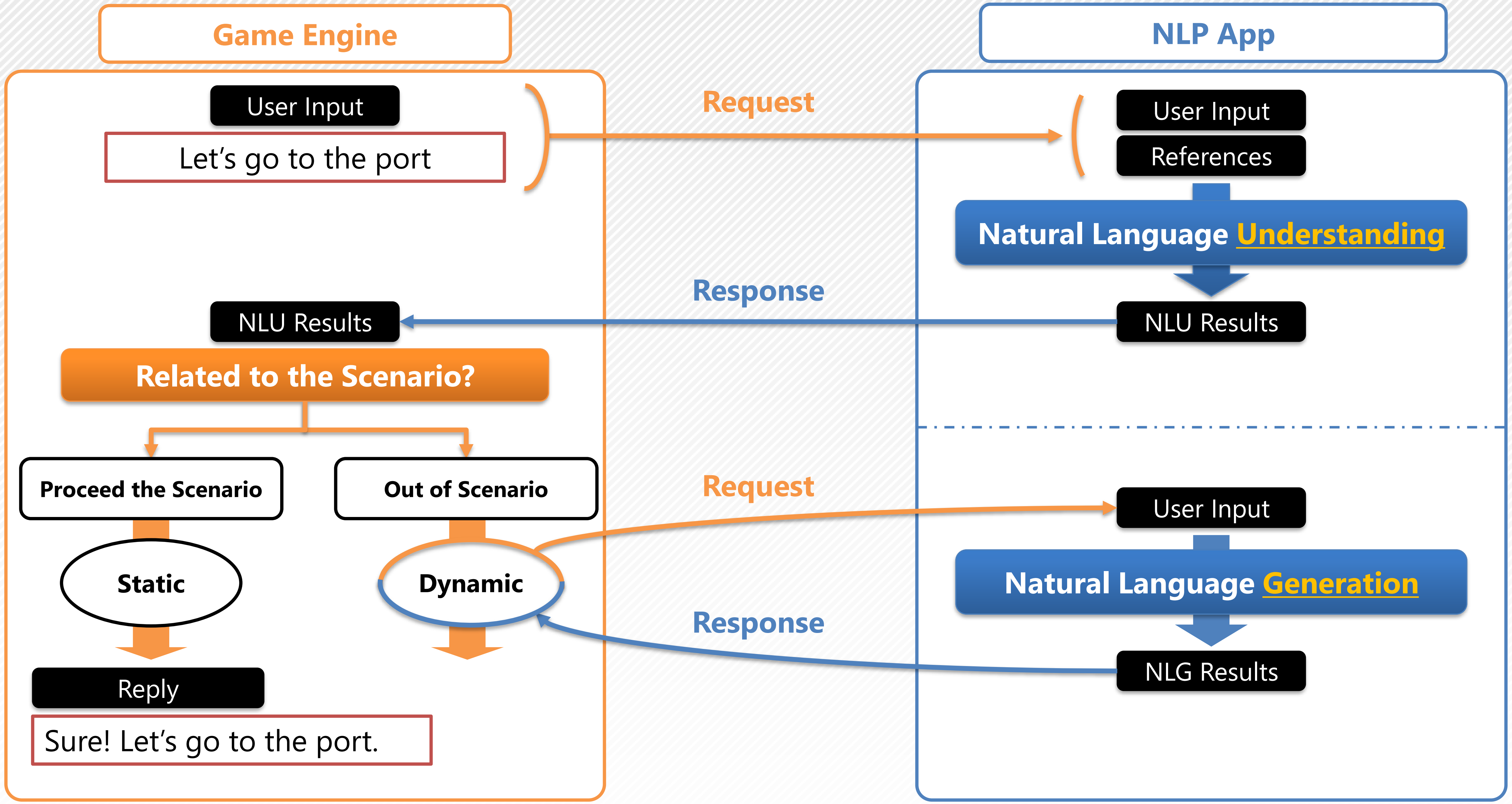
with NLP Adventure

- **“Discovery”**
  - Players discover their route by themselves, without given candidates
  - They don’t have to hear “I cannot understand it” again and again

- **“Sense of immersion”**
  - Input by their own words make players feel immersive into the game

System Overview

System Overview



Game Engine

User Input

Let’s go to the port

NLU Results

Related to the Scenario?

Proceed the Scenario

Out of Scenario

Static

Dynamic

Reply

Sure! Let’s go to the port.

Request

Response

Request

Response

NLP App

User Input

References

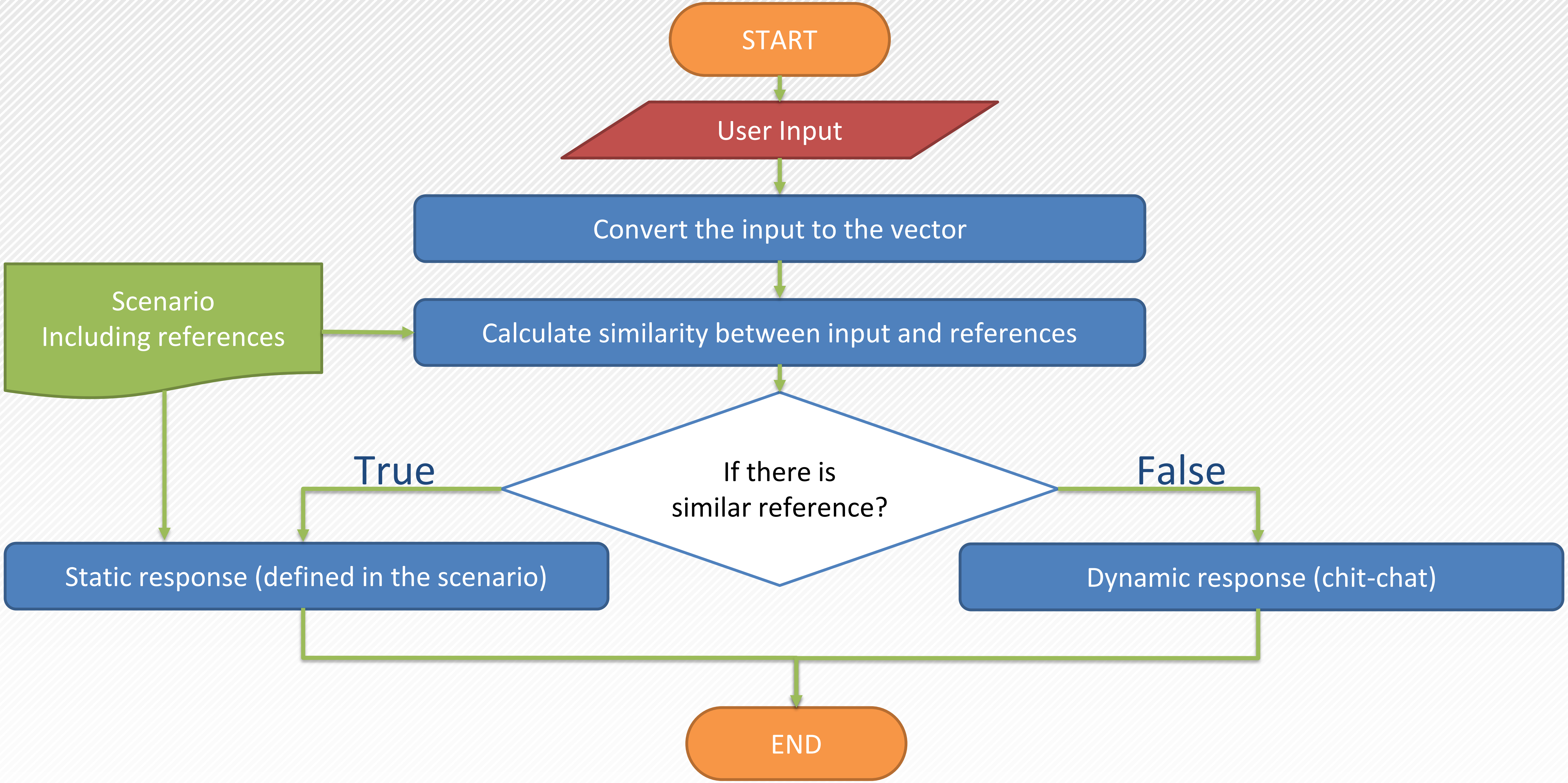
Natural Language **Understanding**

NLU Results

User Input

Natural Language **Generation**

NLG Results



START

User Input

Convert the input to the vector

Scenario Including references

Calculate similarity between input and references

If there is similar reference?

True

False

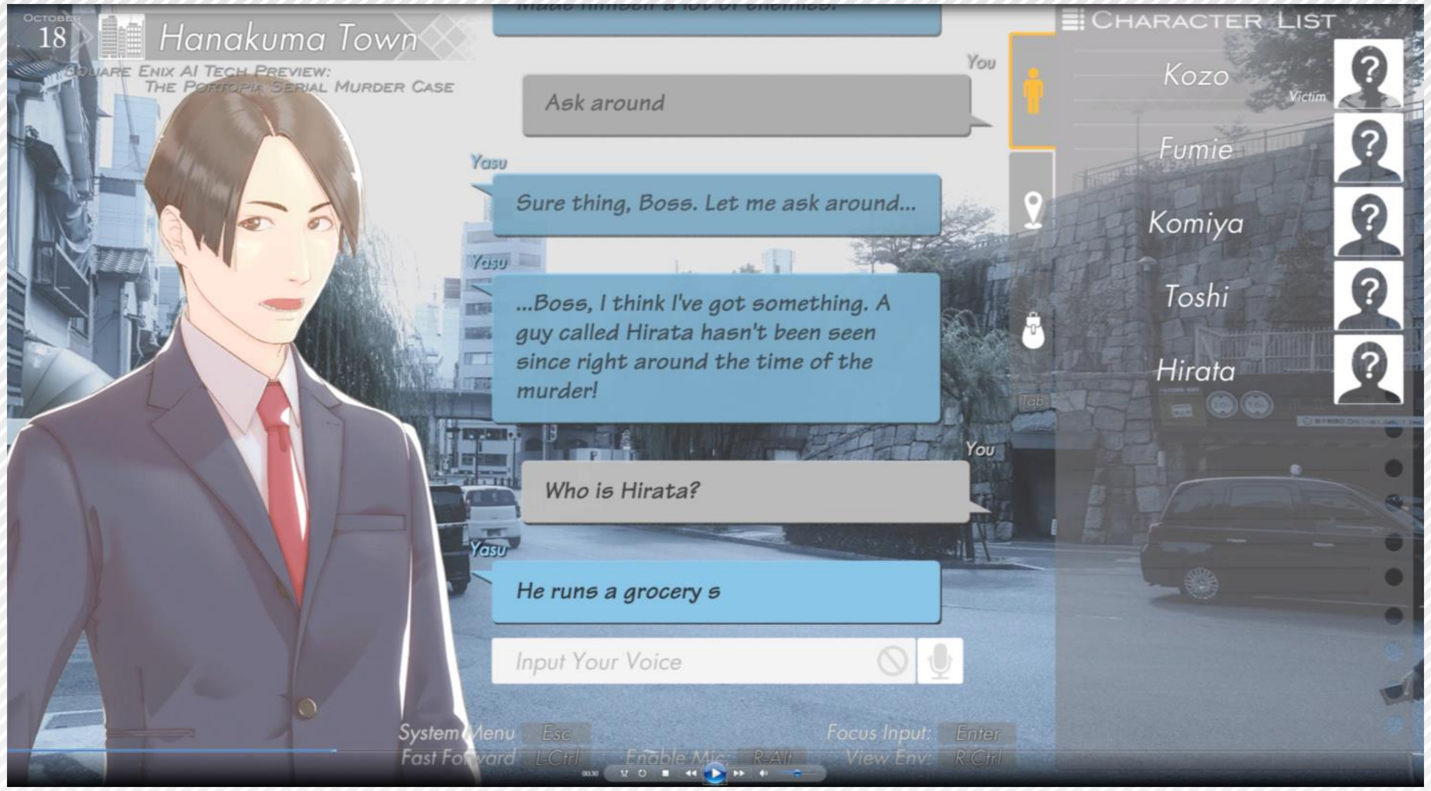
Static response (defined in the scenario)

Dynamic response (chit-chat)

END

Free Input

- Even when users use different words or phrases, the system can **understand users' intent** and continue the scenario



You: Ask around

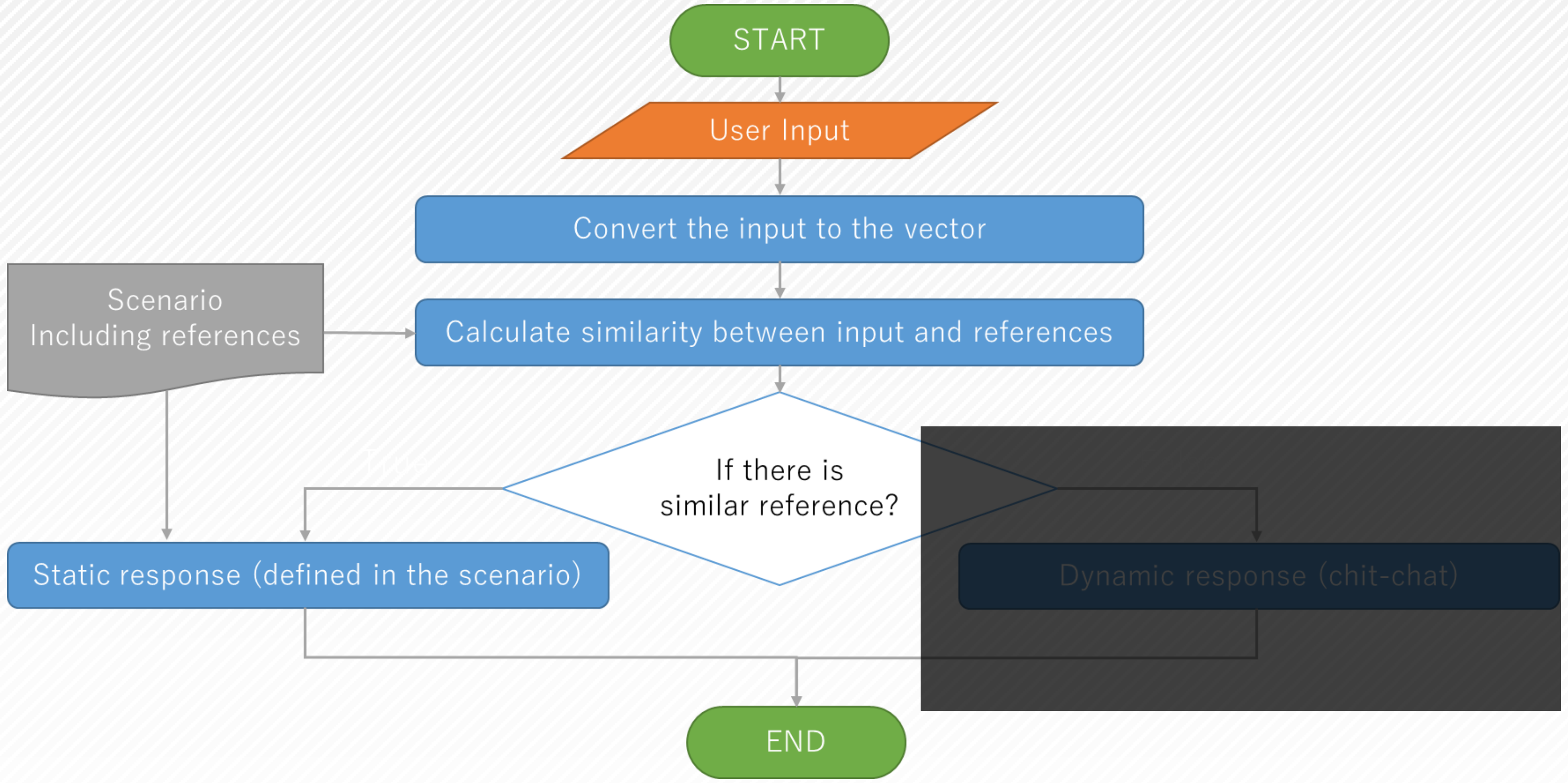
Yasu: Sure thing, Boss. Let me ask around...

Yasu: ...Boss, I think I've got something. A guy called Hirata hasn't been seen since right around the time of the murder!

You: Who is Hirata?

Yasu: He runs a grocery s

Input Your Voice



START

User Input

Convert the input to the vector

Scenario Including references

Calculate similarity between input and references

If there is similar reference?

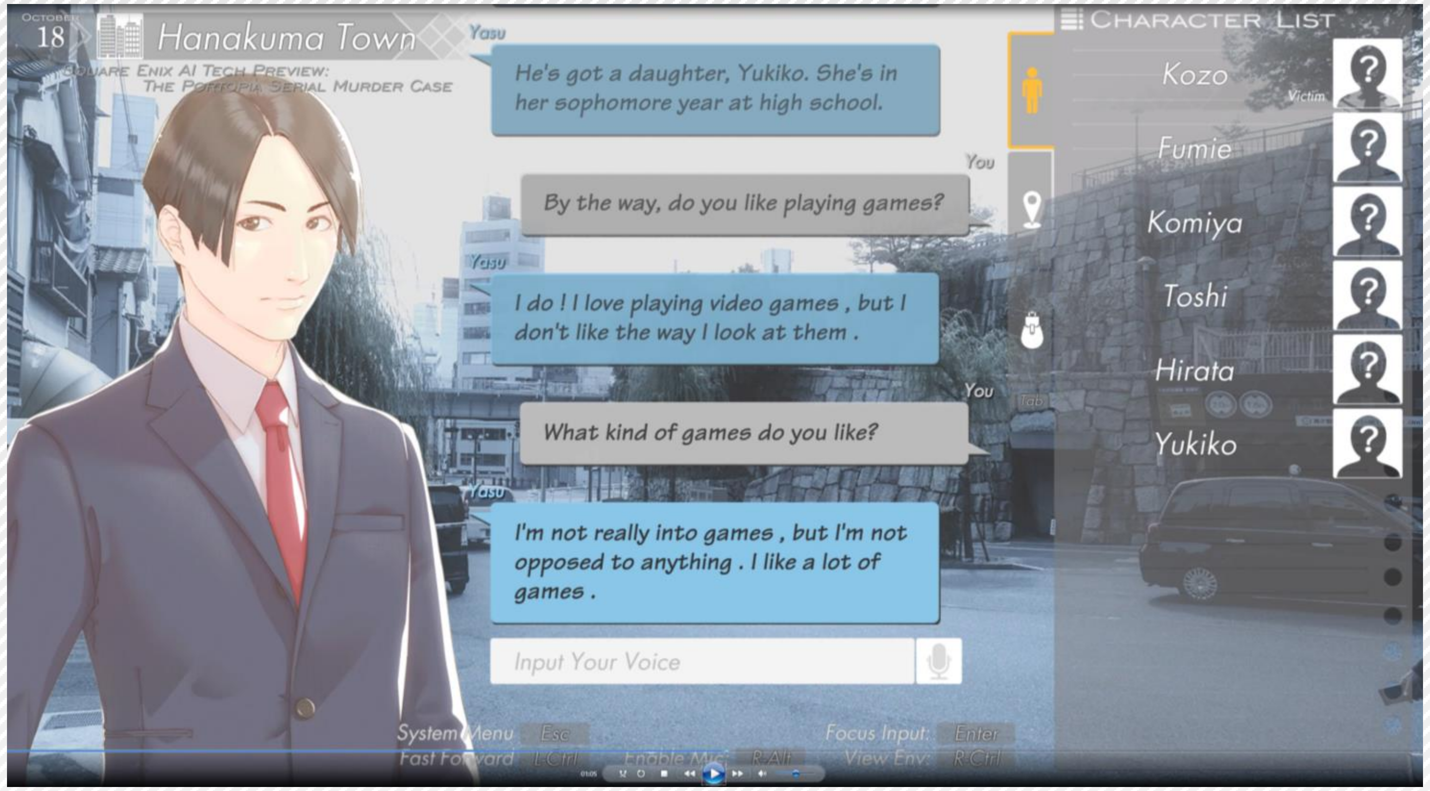
Static response (defined in the scenario)

Dynamic response (chit-chat)

END

Chit-chat

- Even if a player input is unrelated to the story, the character responds to it



Yasu: He's got a daughter, Yukiko. She's in her sophomore year at high school.

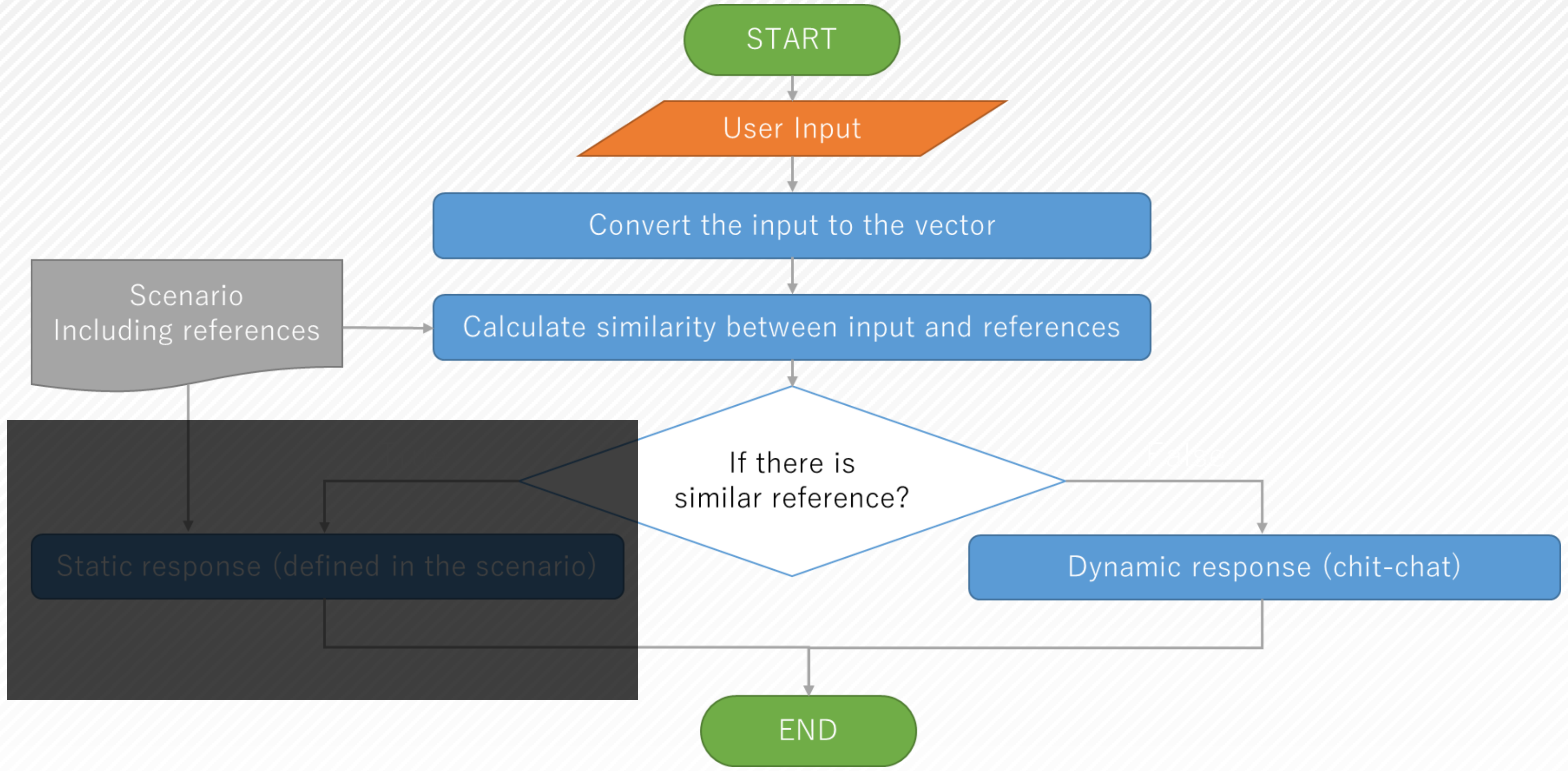
You: By the way, do you like playing games?

Yasu: I do ! I love playing video games , but I don't like the way I look at them .

You: What kind of games do you like?

Yasu: I'm not really into games , but I'm not opposed to anything . I like a lot of games .

Input Your Voice



START

User Input

Convert the input to the vector

Scenario Including references

Calculate similarity between input and references

If there is similar reference?

Static response (defined in the scenario)

Dynamic response (chit-chat)

END

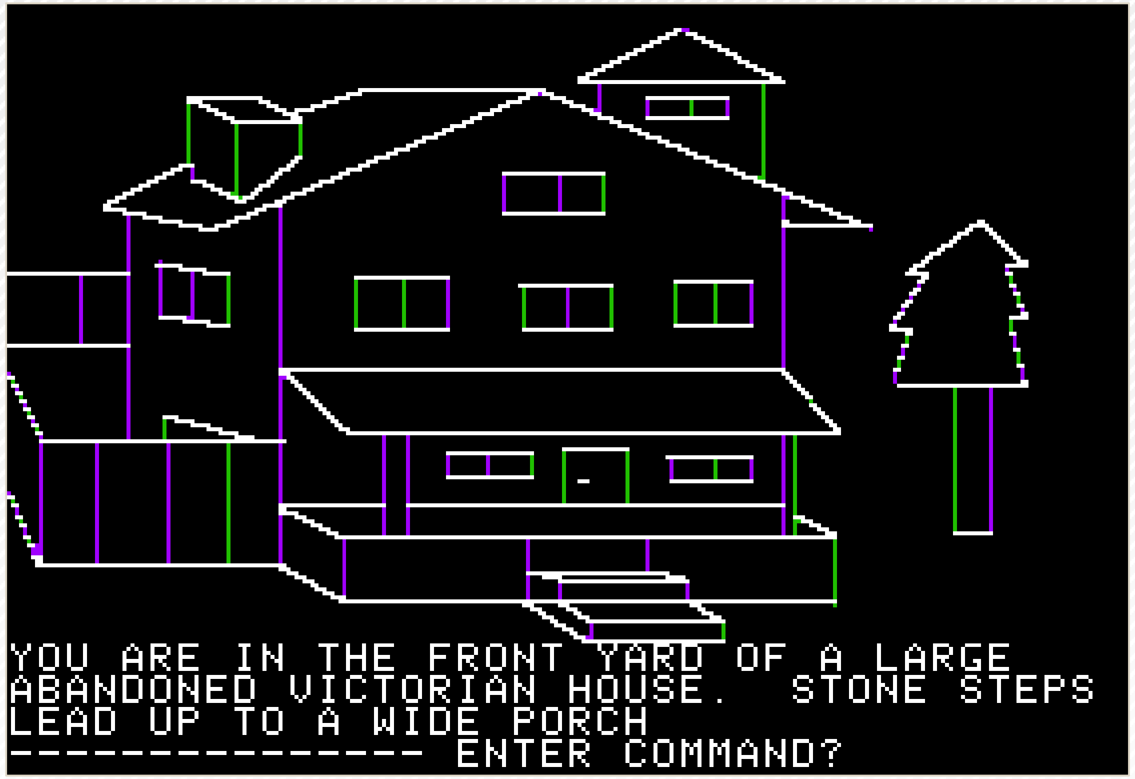
Why now?

Recent Rapid Progress in NLP

Adventure games

In 1980s

- e.g.) Mystery House (1980)
- **“ENTER COMMAND?”**
  - – Use Command Line Interface (CLI) to decide the action



```
YOU ARE IN THE FRONT YARD OF A LARGE
ABANDONED VICTORIAN HOUSE.  STONE STEPS
LEAD UP TO A WIDE PORCH
--------------- ENTER COMMAND?
```

Mystery House (1980) © On-Line Systems
https://upload.wikimedia.org/wikipedia/commons/a/a0/Mystery_House_-_Apple_II_render_emulation_-_2.png

Free Input

How can we evolve adventure games?

- Can't we accommodate more flexible input?
  - **If the meaning is correct, the scenario proceeds even if the input does not exactly match the reference sentence**

“Go to the harbor”, “Head to the harbor”, “Shall we go see the sea?”

↓

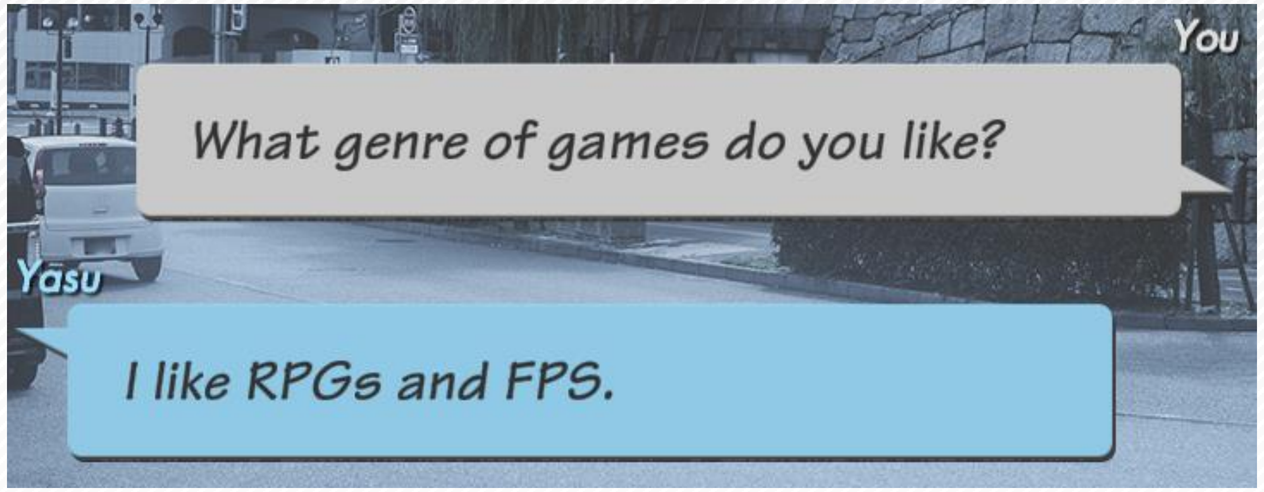
Go to the port

This is what the user intends to do!

Chit-chat

How can we evolve adventure games?

- Don’t show “PLEASE INPUT AGAIN” again and again!
  - **Expect a response even if the input is unrelated to the scenario**



You

What genre of games do you like?

Yasu

I like RPGs and FPS.

© 2023 SQUARE ENIX CO., LTD. All Rights Reserved.

Response is dynamically generated

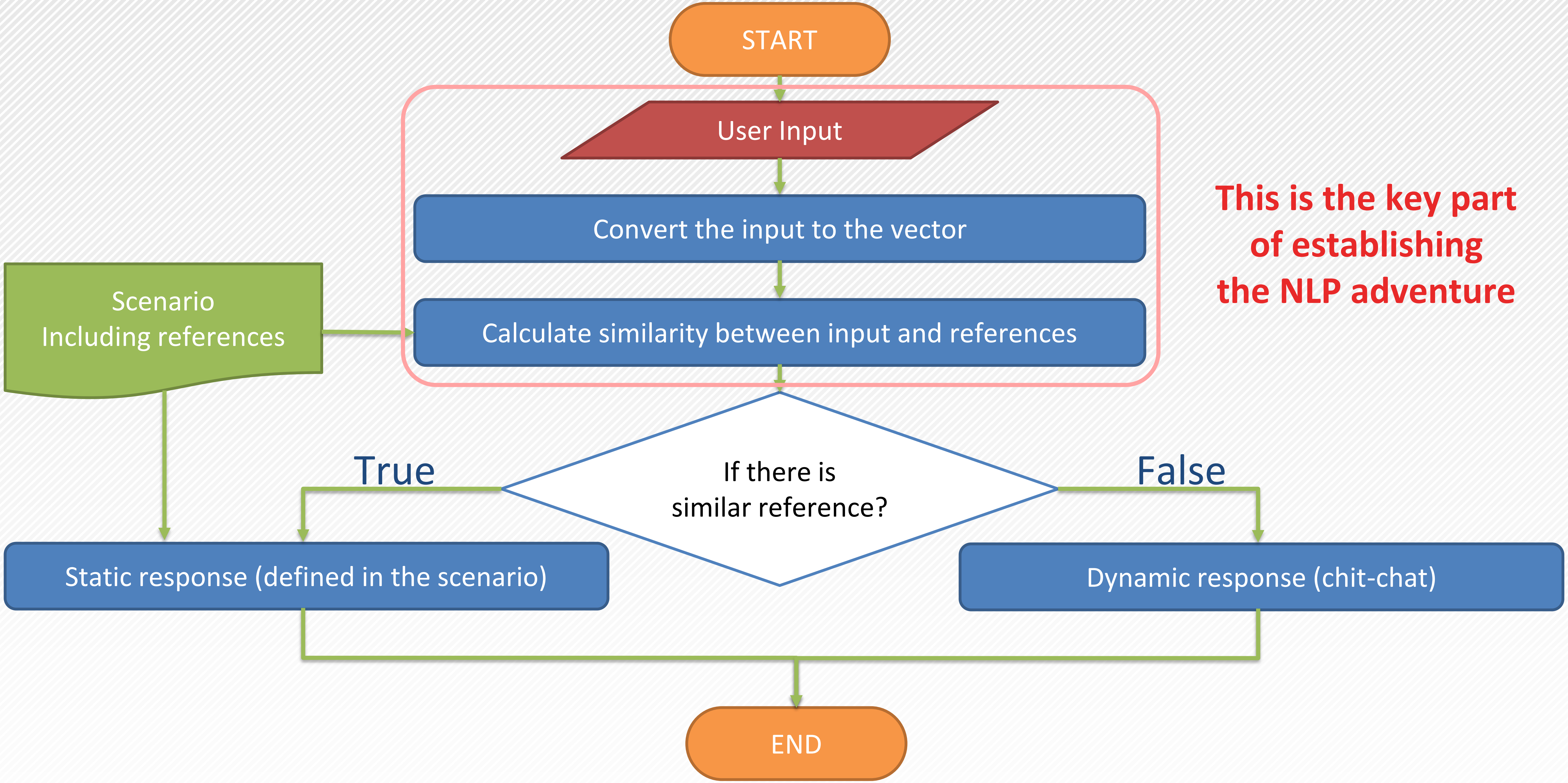
Rapid Progress in NLP

Free Input and Chit-chat can be handled

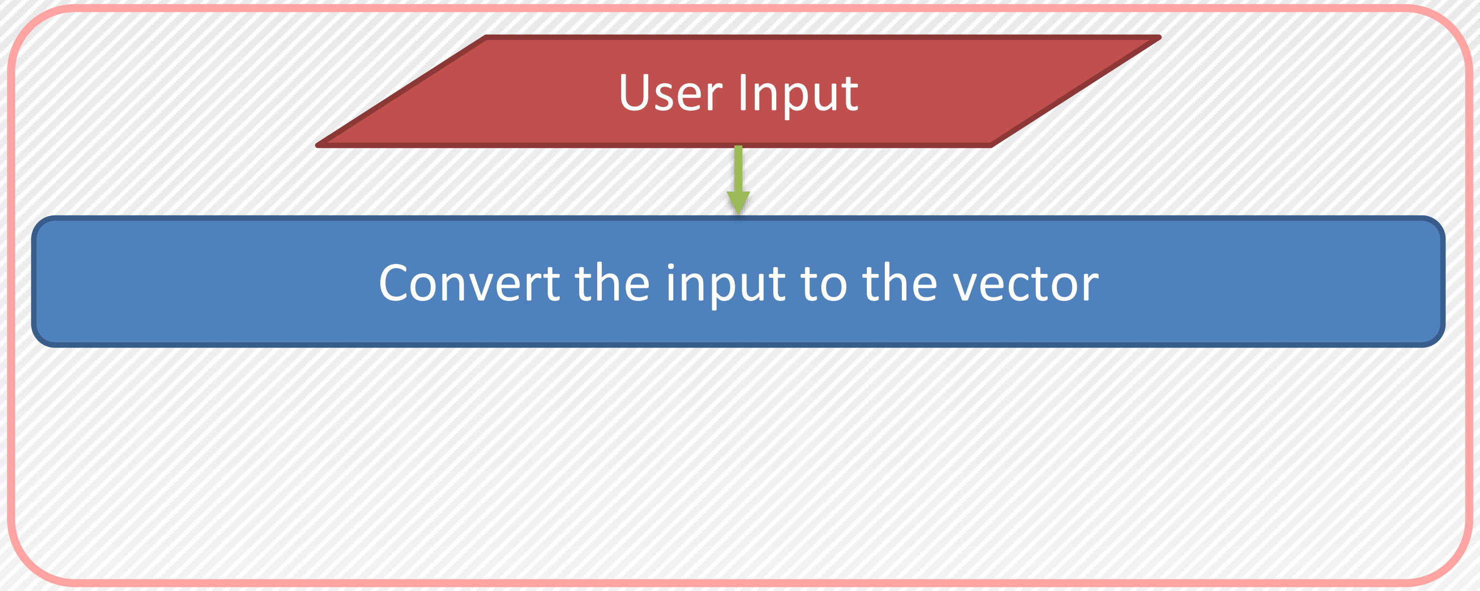
- NLP has come a long way in the last decade.
  - Word/Sentence Embeddings
    - Word2Vec [Mikolov et al., 2013a, 2013b, 2013c] , Skip-Thoughts [Kiros et al., 2015]
  - Transformer [Vaswani et al., 2017]
  - Large-scale Language Models (using Transformer architecture)
    - BERT [Devlin et al., 2019], GPT-3 [Brown et al., 2020]

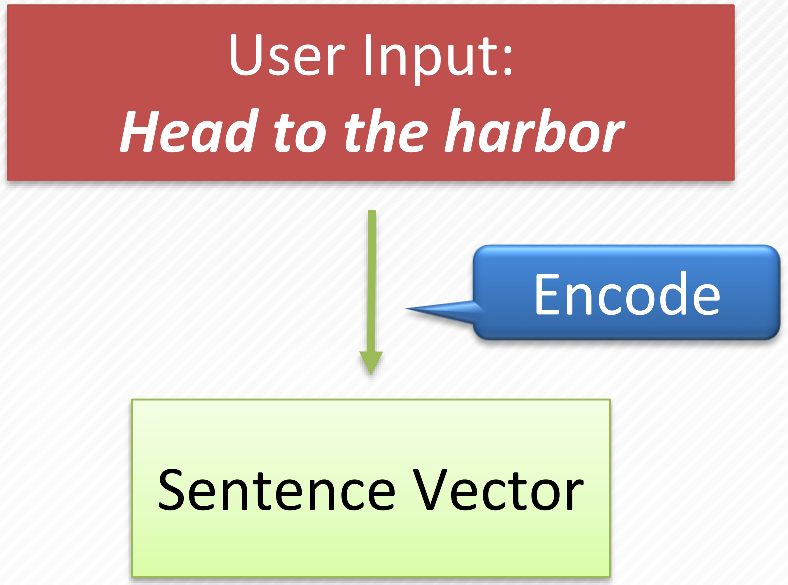
Mikolov et al. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.
Mikolov et al. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of NIPS, 2013.
Mikolov et al. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of NAACL-HLT, 2013.
Kiros et al. Skip-Thought Vectors. In Proceedings of NIPS, 2015.
Vaswani et al. Attention is All you Need. In Proceedings of NIPS, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of NAACL-HLT, 2019.
Brown et al. Language Models are Few-Shot Learners. In Proceedings of NeurIPS, 2020.

Flow of “NLP Adventure” System

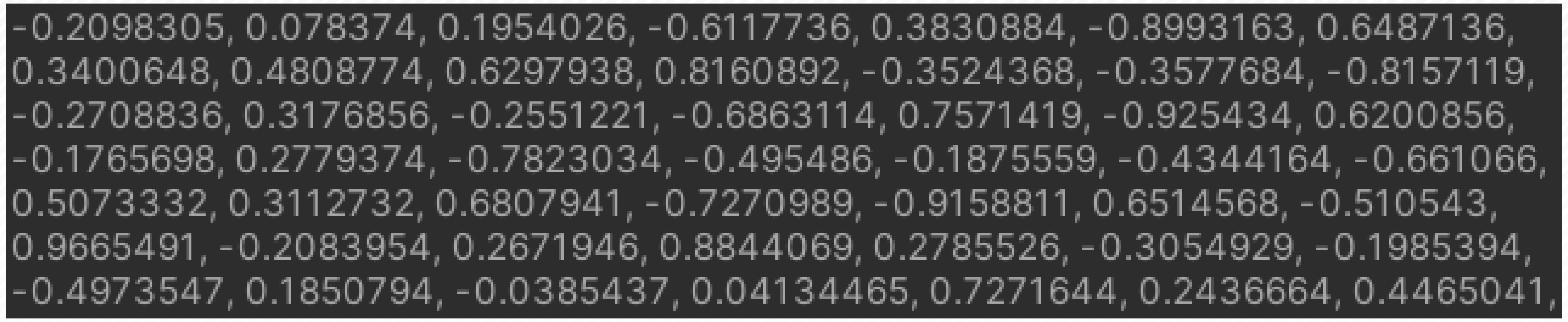

START
User Input
Convert the input to the vector
Scenario
Including references
Calculate similarity between input and references
True
If there is
similar reference?
False
Static response (defined in the scenario)
Dynamic response (chit-chat)
END

This is the key part of establishing the NLP adventure

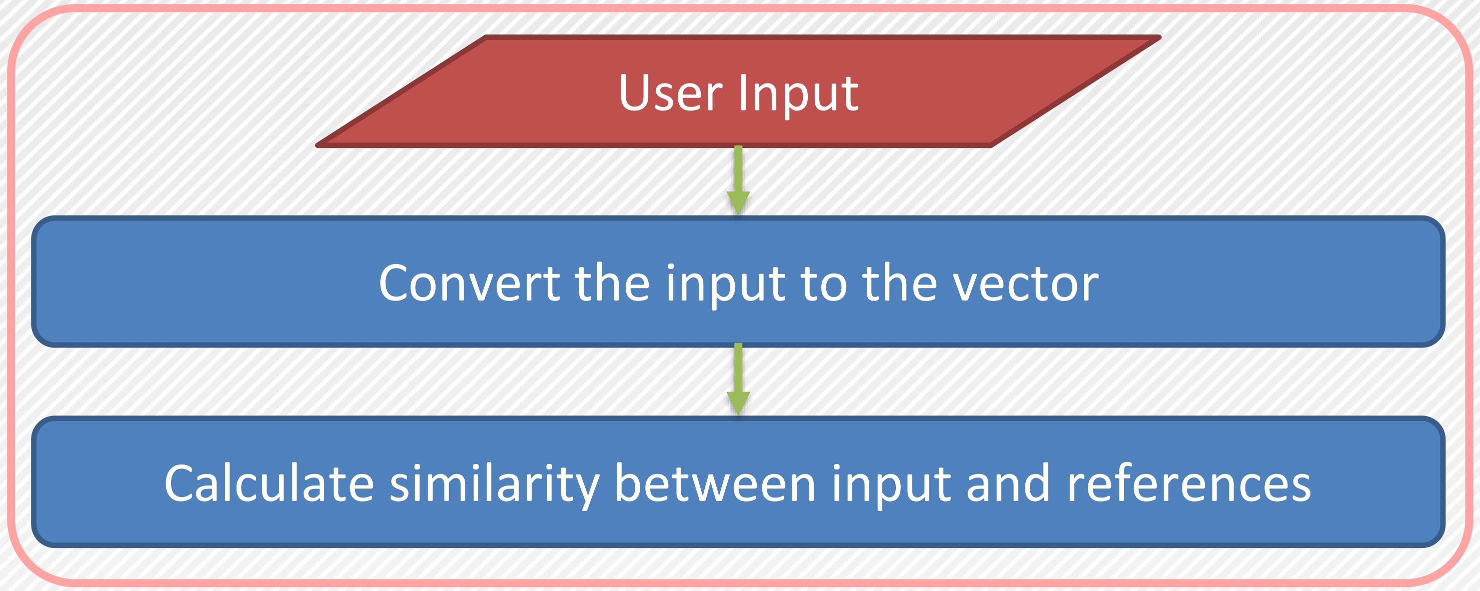




a numerical representation of the meaning of a sentence
(this is an excerpt, it actually has many more dimensions)



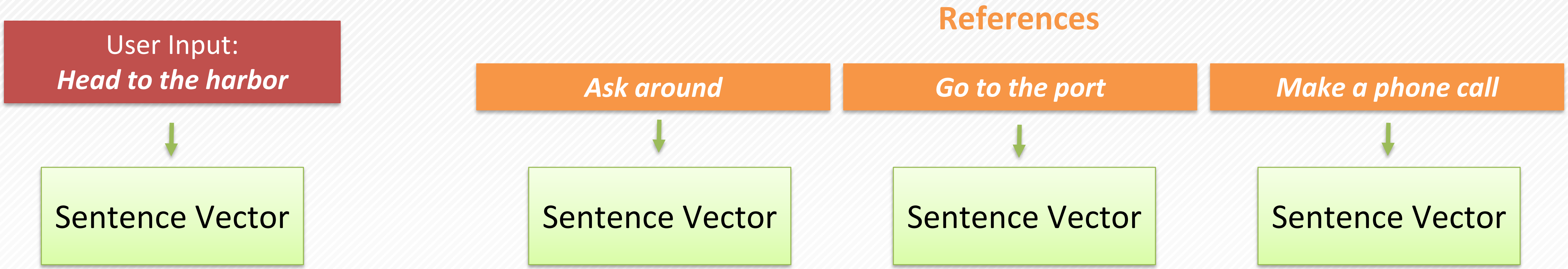
```
-0.2098305, 0.078374, 0.1954026, -0.6117736, 0.3830884, -0.8993163, 0.6487136,
0.3400648, 0.4808774, 0.6297938, 0.8160892, -0.3524368, -0.3577684, -0.8157119,
-0.2708836, 0.3176856, -0.2551221, -0.6863114, 0.7571419, -0.925434, 0.6200856,
-0.1765698, 0.2779374, -0.7823034, -0.495486, -0.1875559, -0.4344164, -0.661066,
0.5073332, 0.3112732, 0.6807941, -0.7270989, -0.9158811, 0.6514568, -0.510543,
0.9665491, -0.2083954, 0.2671946, 0.8844069, 0.2785526, -0.3054929, -0.1985394,
-0.4973547, 0.1850794, -0.0385437, 0.04134465, 0.7271644, 0.2436664, 0.4465041,
```



User Input

Convert the input to the vector

Calculate similarity between input and references



User Input:
Head to the harbor

Sentence Vector

References

Ask around

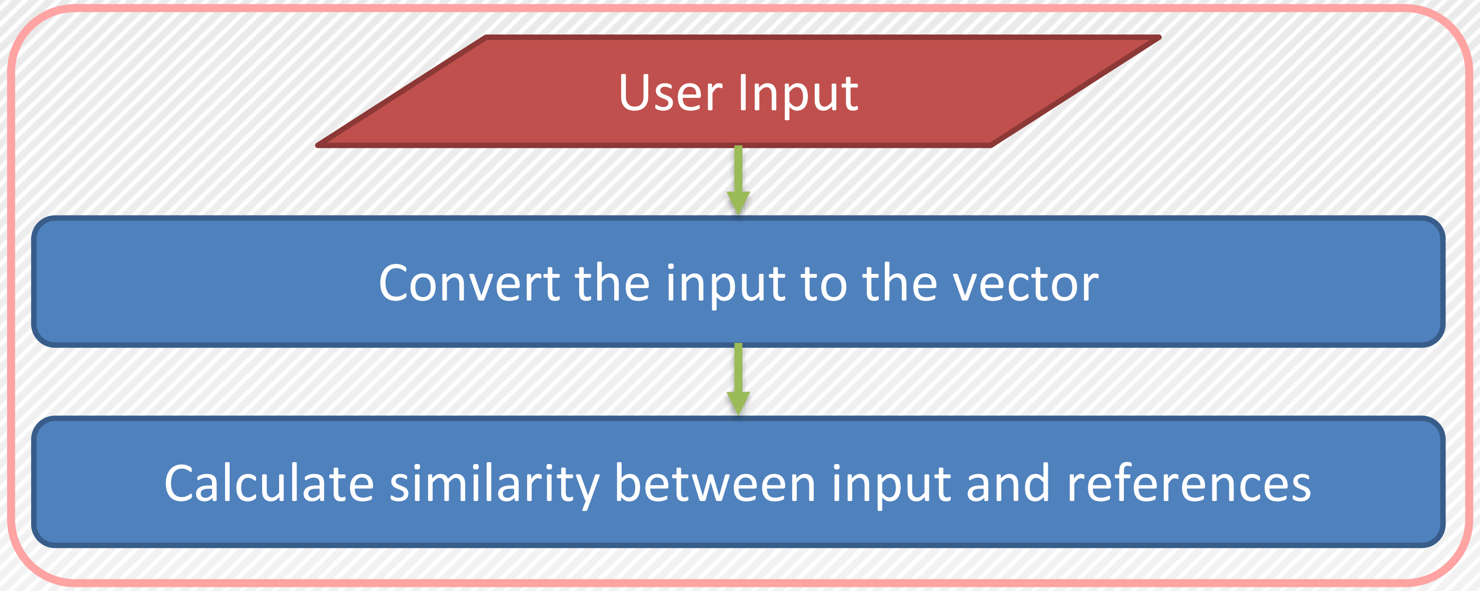
Go to the port

Make a phone call

Sentence Vector

Sentence Vector

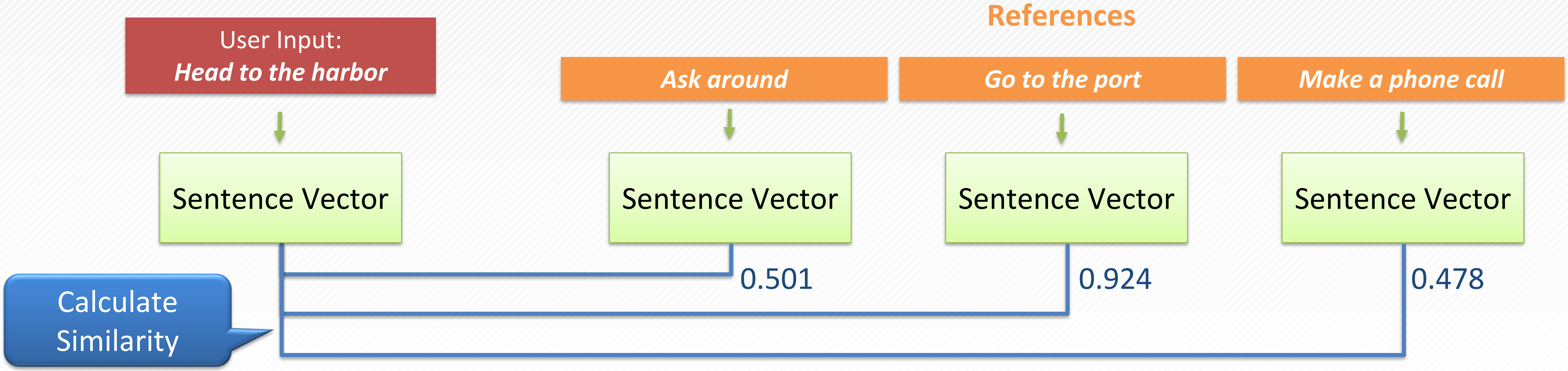
Sentence Vector



User Input

Convert the input to the vector

Calculate similarity between input and references



User Input:
Head to the harbor

References

Ask around

Go to the port

Make a phone call

Sentence Vector

Sentence Vector

Sentence Vector

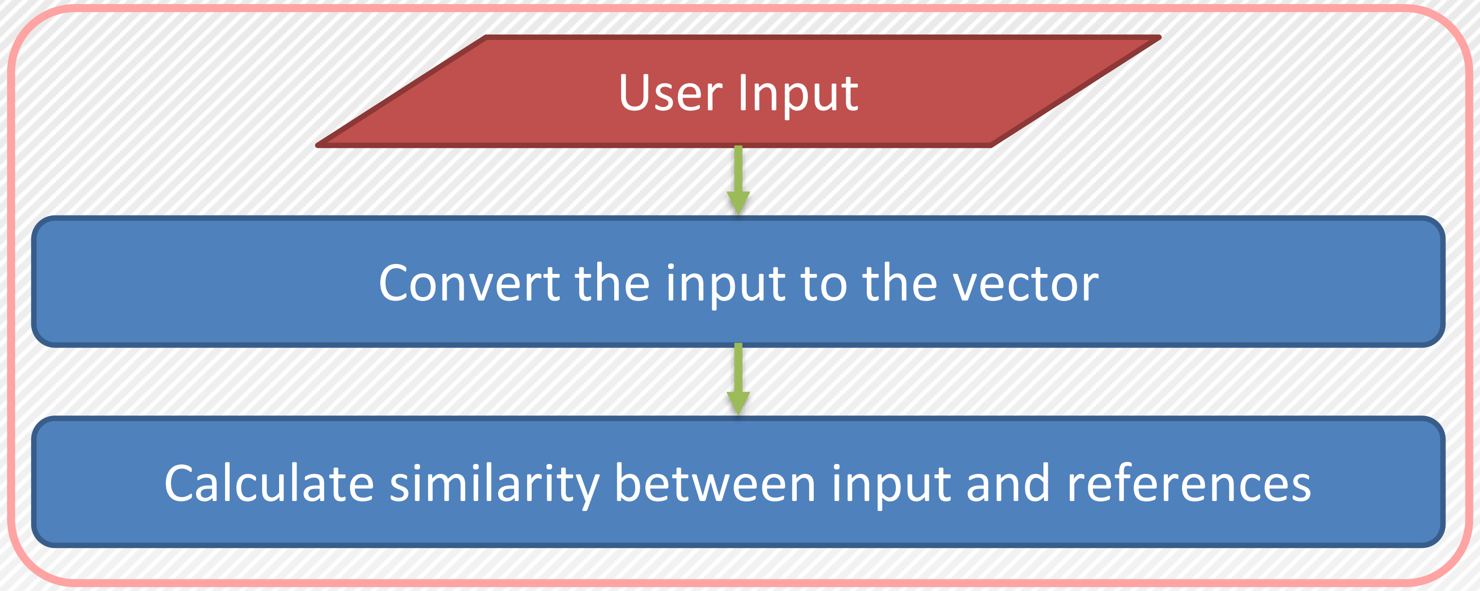
Sentence Vector

0.501

0.924

0.478

Calculate Similarity



User Input

Convert the input to the vector

Calculate similarity between input and references

References	Similarity
Ask around	0.501
Go to the port	0.924
Make a phone call	0.478
Go to Kozo's house	0.604
Get back to Headquarters	0.645

This is what the user intends to do!

User Input:
Head to the harbor

Sentence Vector

For predicting the user's intent

- Embed a sentence into a vector space
  - Skip-Thoughts
  - Transformer-based technologies today:
    - Sentence-BERT [Reimers & Gurevych, 2019]
    - SimCSE [Gao et al., 2021]

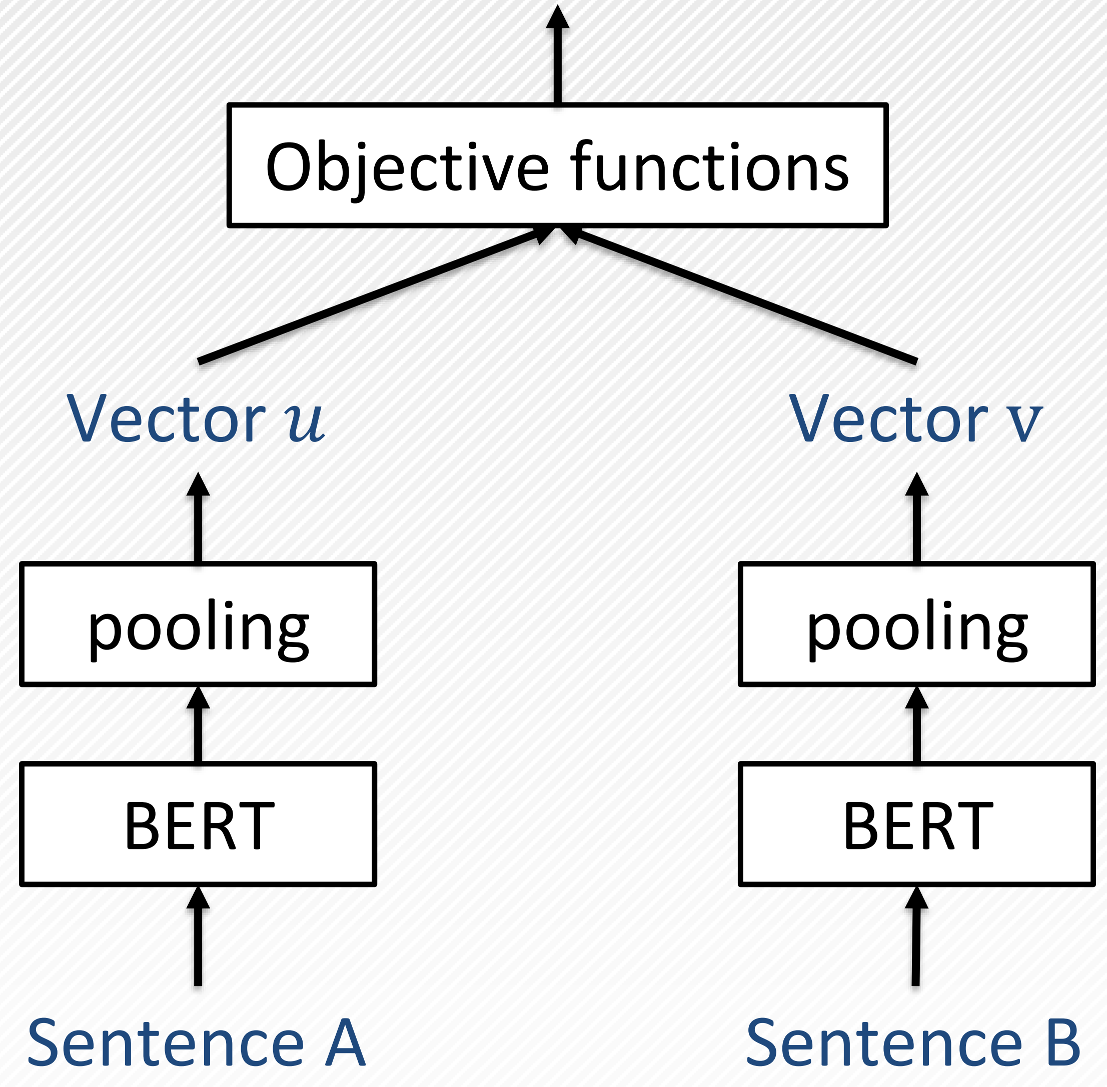
- Then, numerical calculations can be done
  - Cosine Similarity

Reimers & Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In Proceedings of EMNLP-IJCNLP, 2019.
Gao et al. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In Proceedings of EMNLP, 2021.

Sentence Vector

Example

- Sentence-BERT (as an Example)
  - https://www.sbert.net/
  - Use paired sentences for training
  - Trained model can convert sentences to vectors with similar representations if the sentences have similar meanings

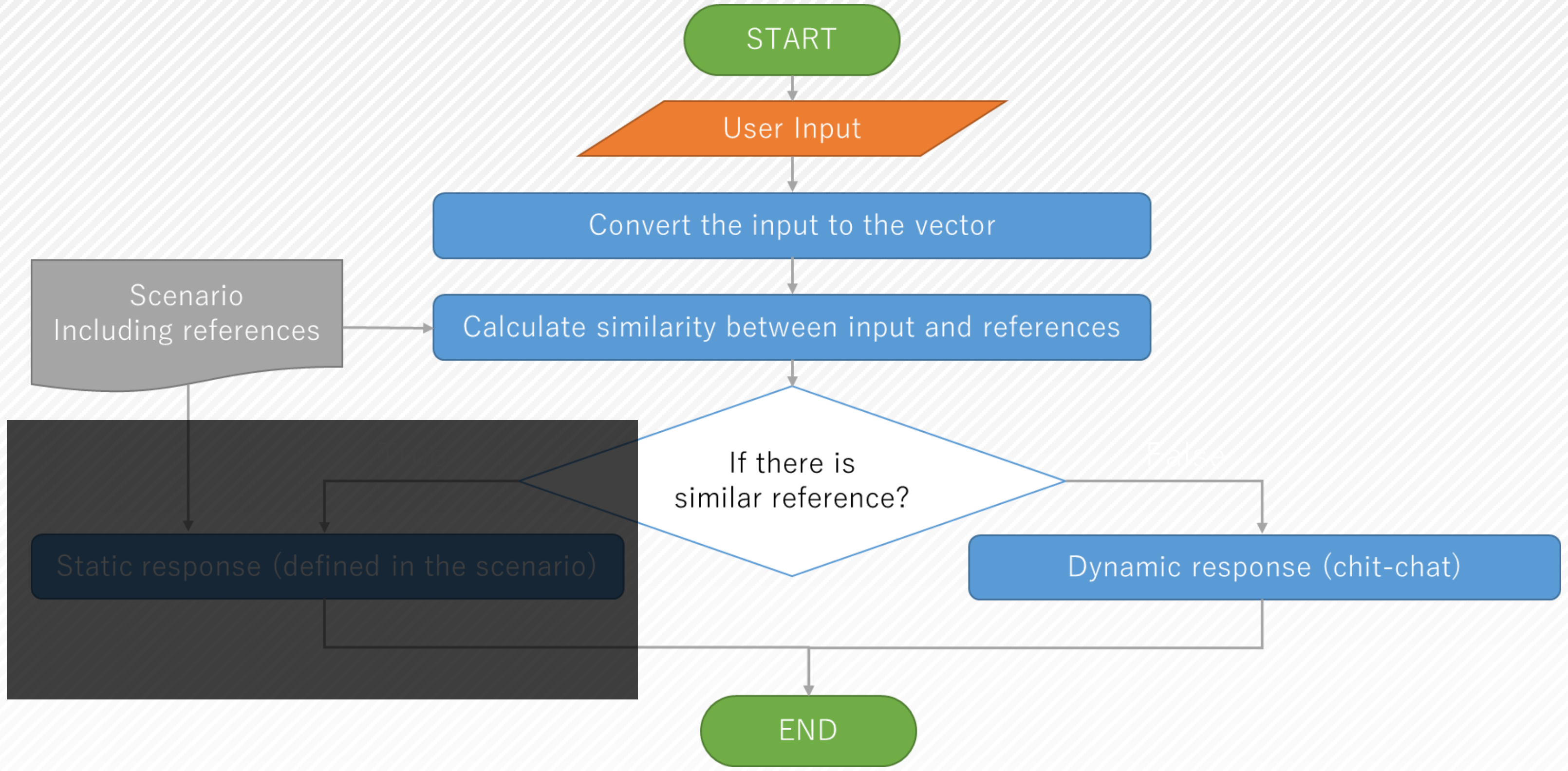


This figure is based on Figure 1 & 2 from [Reimers & Gurevych, 2019]

Chit-chat Dialogue

- Outside of the main game flow, responses are generated dynamically

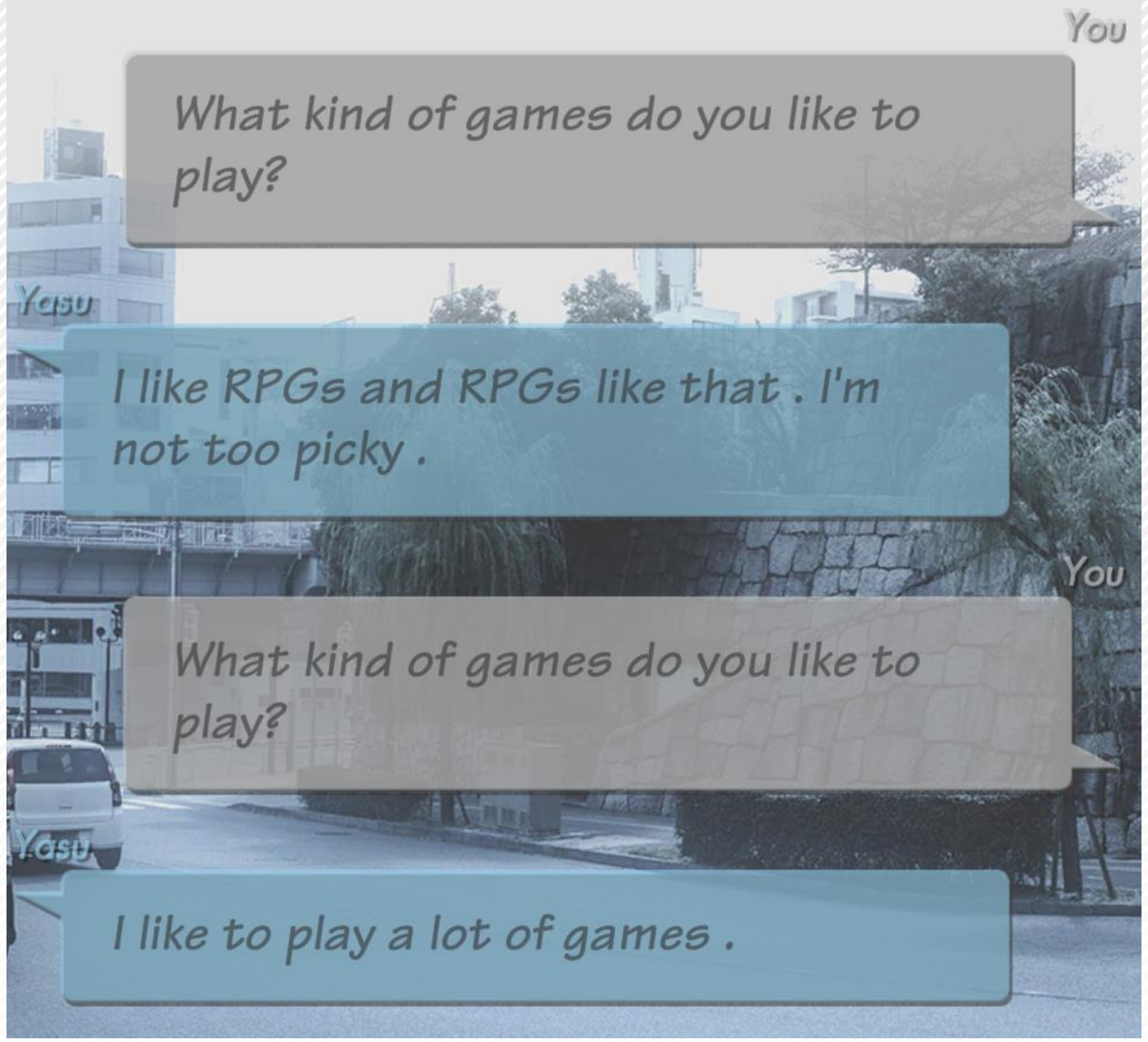
- General chat bot models
  - DialoGPT [Zhang et al., 2020]
  - BlenderBot [Roller et al., 2021]


START
User Input
Convert the input to the vector
Scenario
Including references
Calculate similarity between input and references
If there is
similar reference?
False
Static response (defined in the scenario)
Dynamic response (chit-chat)
END

Zhang et al. DIALOGPT : Large-Scale Generative Pre-training for Conversational Response Generation. In Proceedings of ACL, 2020.
Roller et al. Recipes for Building an Open-Domain Chatbot. In Proceedings of EACL, 2021.

Chit-chat Dialogue

The Same Question, Different Answers



You

What kind of games do you like to play?

Yasu

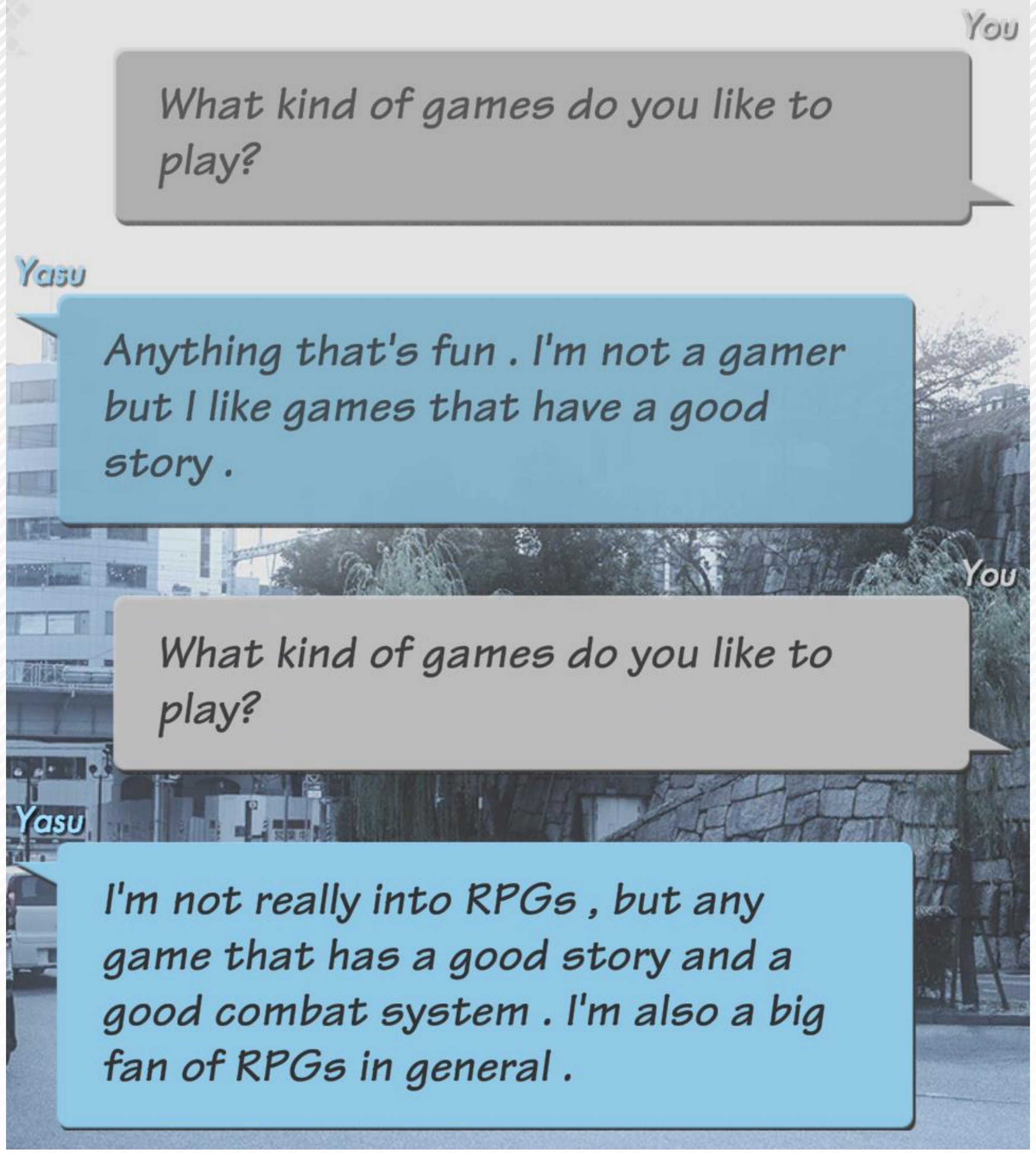
I like RPGs and RPGs like that . I'm not too picky .

You

What kind of games do you like to play?

Yasu

I like to play a lot of games .



You

What kind of games do you like to play?

Yasu

Anything that's fun . I'm not a gamer but I like games that have a good story .

You

What kind of games do you like to play?

Yasu

I'm not really into RPGs , but any game that has a good story and a good combat system . I'm also a big fan of RPGs in general .

Chit-chat Dialogue

Issues to be solved

- Some of the issues with text generation
  - How to prevent repetition?
  - How does character knowledge become persistent?
  - How much context that AI needs to know to respond?

Chit-chat Dialogue

Future Work

- Reflect the persona/knowledge of each character

- NOTE:
  NLP should be based on the game world, not our real world
  - General NLP methods are mainly designed for the real world, not for the fictional world
  - We can use the techniques of real-world NLP,
    but should develop new techniques: **NLP for game worlds**

How to Implement?

Free Input and Chit-chat by NLP for Games

- **To be embedded into a game!**

- We first started implementation from the aforementioned “server-client” approach.
- However, we would rather contain NLP apps into the game.

Client-Server to One-Application

Avoiding Client-Server

Why?

- Current NLP applications are mainly uses client-server approach
- However, if we choose it, there are multiple disadvantages

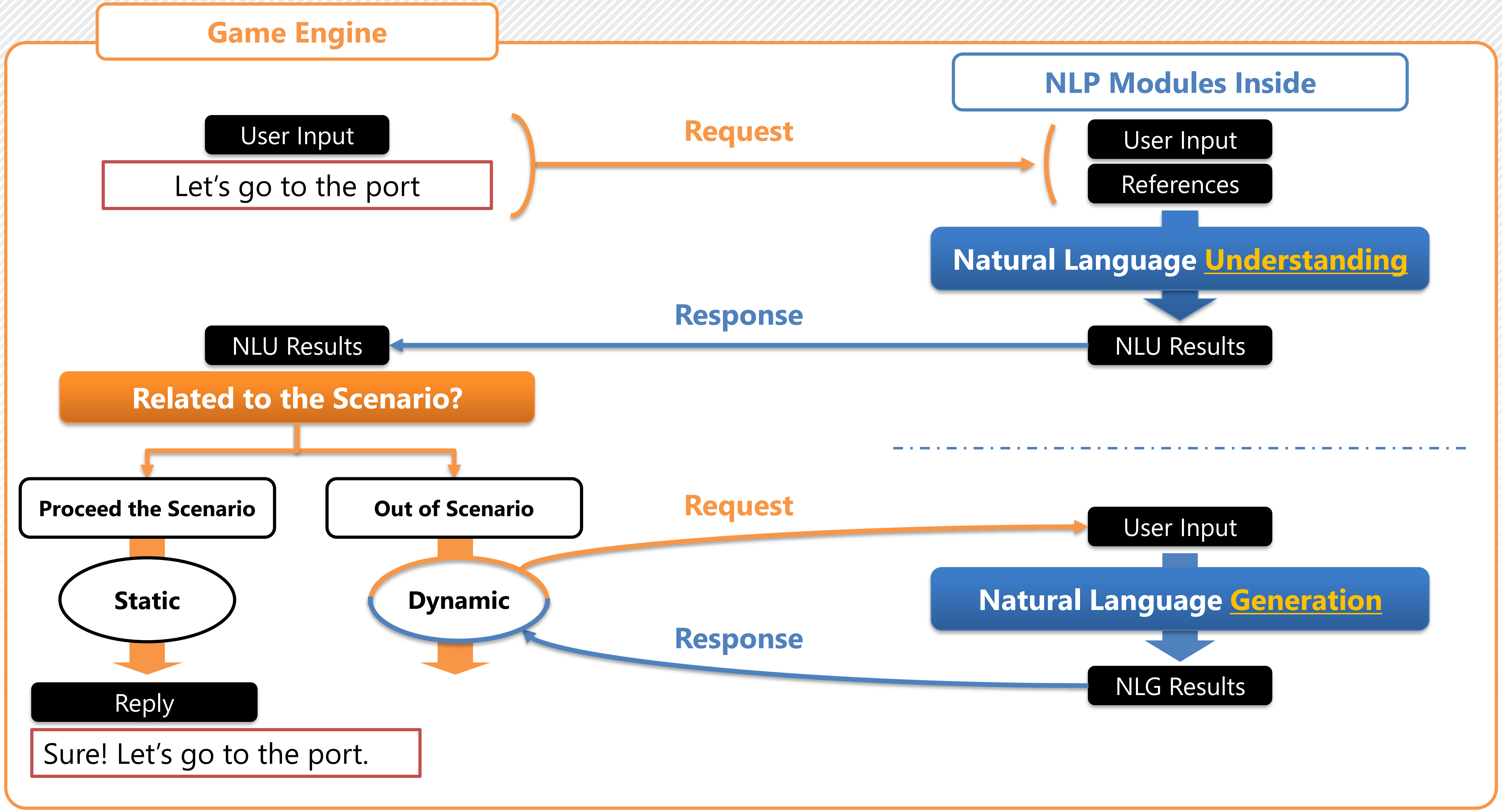
Problems in Client-Server

- Players can’t play the game without the Internet connection

- We should maintain the NLP server
  - When we stop the server, all the games will be stopped

- All the inputs are gathered by the publisher
  - Publisher will face the security risk to treat the data safely

“One application” Overview



How to implement it?

NLP App in Game Engines

- In this deep learning era, research in NLP field is tend to be conducted with Python as main programming language

- How to solve the problem of embedding them in Game Engines?

ONNX

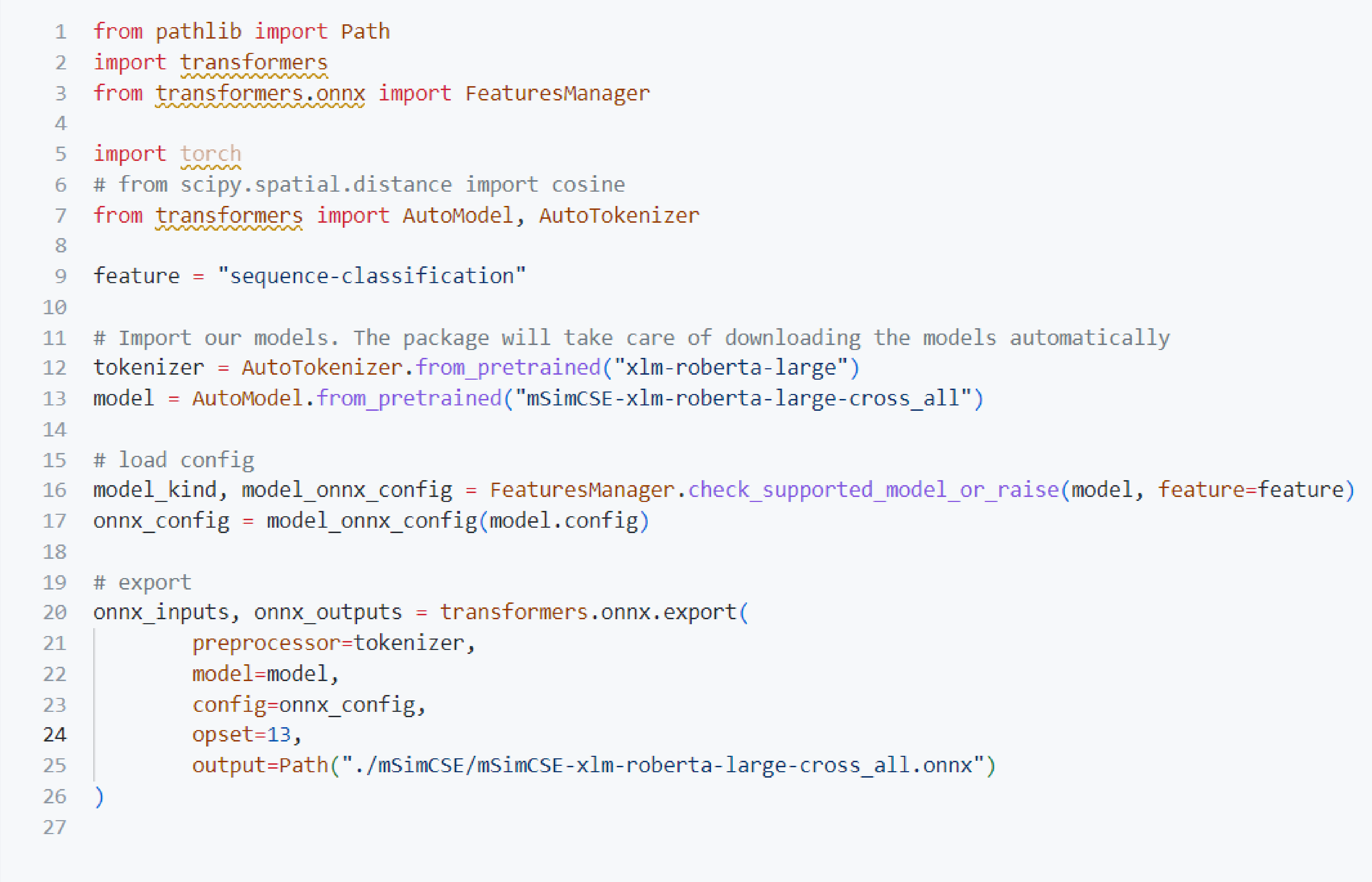
- Convert models to **“ONNX”** format to handle them in various language
  - To be better handled in Game Engines

- There are a lot of useful information and tools on ONNX conversion of language models
  - Natural Language Understanding (NLU) model for “Free Input”
  - Natural Language Generation (NLG) model for “Chit-chat”

Convert models to ONNX

For NLU

- BERT-like models
  (Transformer Encoder-based)

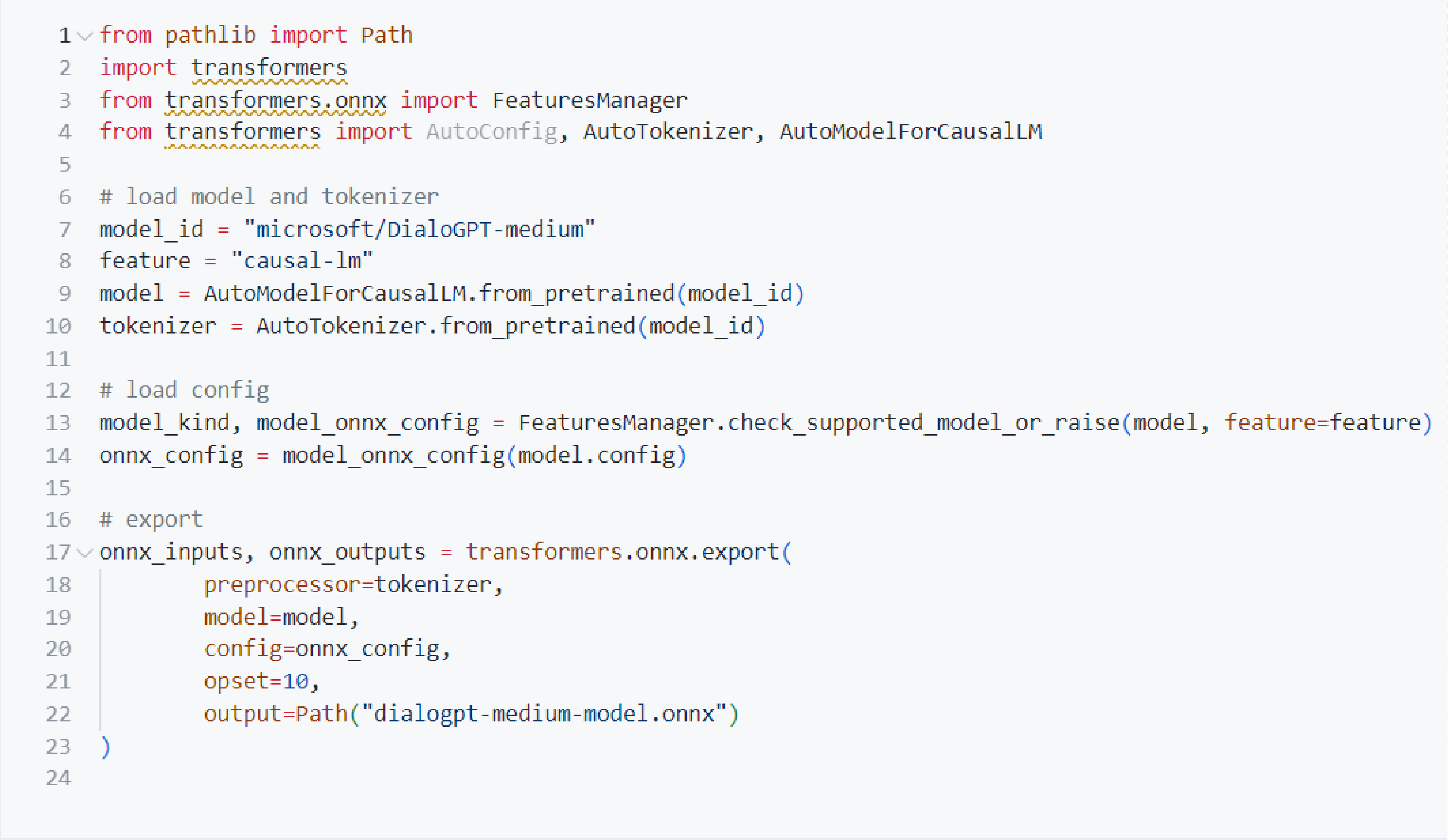


```python
from pathlib import Path
import transformers
from transformers.onnx import FeaturesManager

import torch
# from scipy.spatial.distance import cosine
from transformers import AutoModel, AutoTokenizer

feature = "sequence-classification"

# Import our models. The package will take care of downloading the models automatically
tokenizer = AutoTokenizer.from_pretrained("xlm-roberta-large")
model = AutoModel.from_pretrained("mSimCSE-xlm-roberta-large-cross_all")

# load config
model_kind, model_onnx_config = FeaturesManager.check_supported_model_or_raise(model, feature=feature)
onnx_config = model_onnx_config(model.config)

# export
onnx_inputs, onnx_outputs = transformers.onnx.export(
        preprocessor=tokenizer,
        model=model,
        config=onnx_config,
        opset=13,
        output=Path("./mSimCSE/mSimCSE-xlm-roberta-large-cross_all.onnx")
)
```

Convert models to ONNX

For NLG

- GPT-like models
  (Transformer Decoder-based)



```python
from pathlib import Path
import transformers
from transformers.onnx import FeaturesManager
from transformers import AutoConfig, AutoTokenizer, AutoModelForCausalLM

# load model and tokenizer
model_id = "microsoft/DialoGPT-medium"
feature = "causal-lm"
model = AutoModelForCausalLM.from_pretrained(model_id)
tokenizer = AutoTokenizer.from_pretrained(model_id)

# load config
model_kind, model_onnx_config = FeaturesManager.check_supported_model_or_raise(model, feature=feature)
onnx_config = model_onnx_config(model.config)

# export
onnx_inputs, onnx_outputs = transformers.onnx.export(
        preprocessor=tokenizer,
        model=model,
        config=onnx_config,
        opset=10,
        output=Path("dialogpt-medium-model.onnx")
)

```

Tokenizers required

- Now, you have ONNX formatted models

- BUT, you have to prepare tokenizers to run models

Preparing Tokenizers

- What’s Tokenizers?
  – Split a sentence into tokens, then convert them to numbers
  – Converting data to numbers so that they can be handled by machine learning models

```
Sample sentence: Do you like playing adventure games?
Tokenized      : ['_Do', '_you', '_like', '_playing', '_adventure', '_games', '?']
Encoded        : [0, 984, 398, 1884, 75169, 156578, 27528, 32, 2]
Decoded        : <s> Do you like playing adventure games?</s>
```

Preparing Tokenizers

- Run the tokenizer corresponding to the model
  - Word level Tokenizer
  - Character level Tokenizer
  - Sub-word level Tokenizer
    - BPE
    - Unigram
    - Sentencepiece

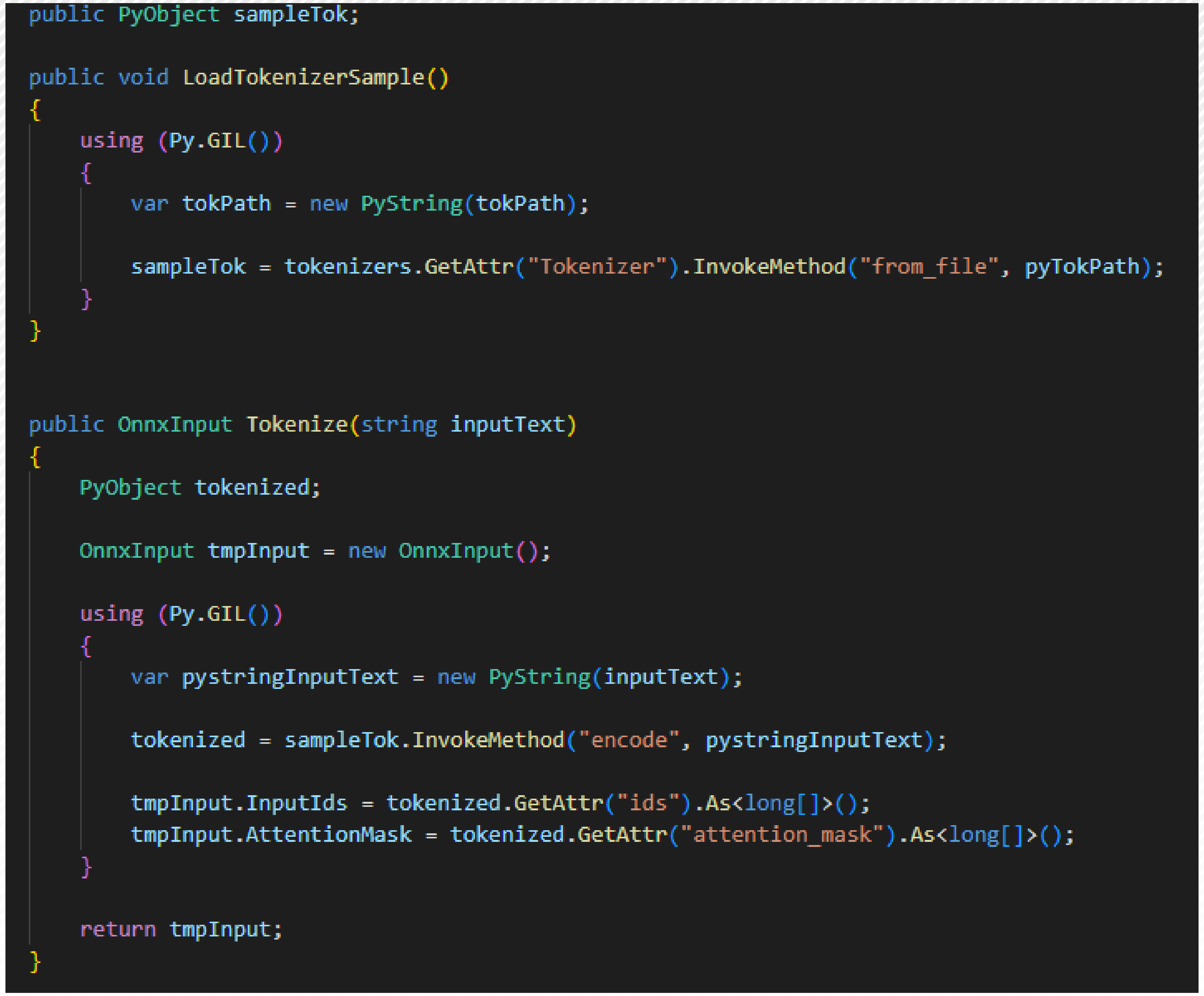
Preparing Tokenizers

- Convert a tokenizer
  - From Hugging Face Transformers' style to Hugging Face Tokenizers'
  - [TIPS for implementation]
    - We use HF Tokenizers instead of HF Transformers to avoid logger function
    - Game application may not have "console" to handle Python's stdout/stderr, so logging function causes error

Use Required Tokenizer

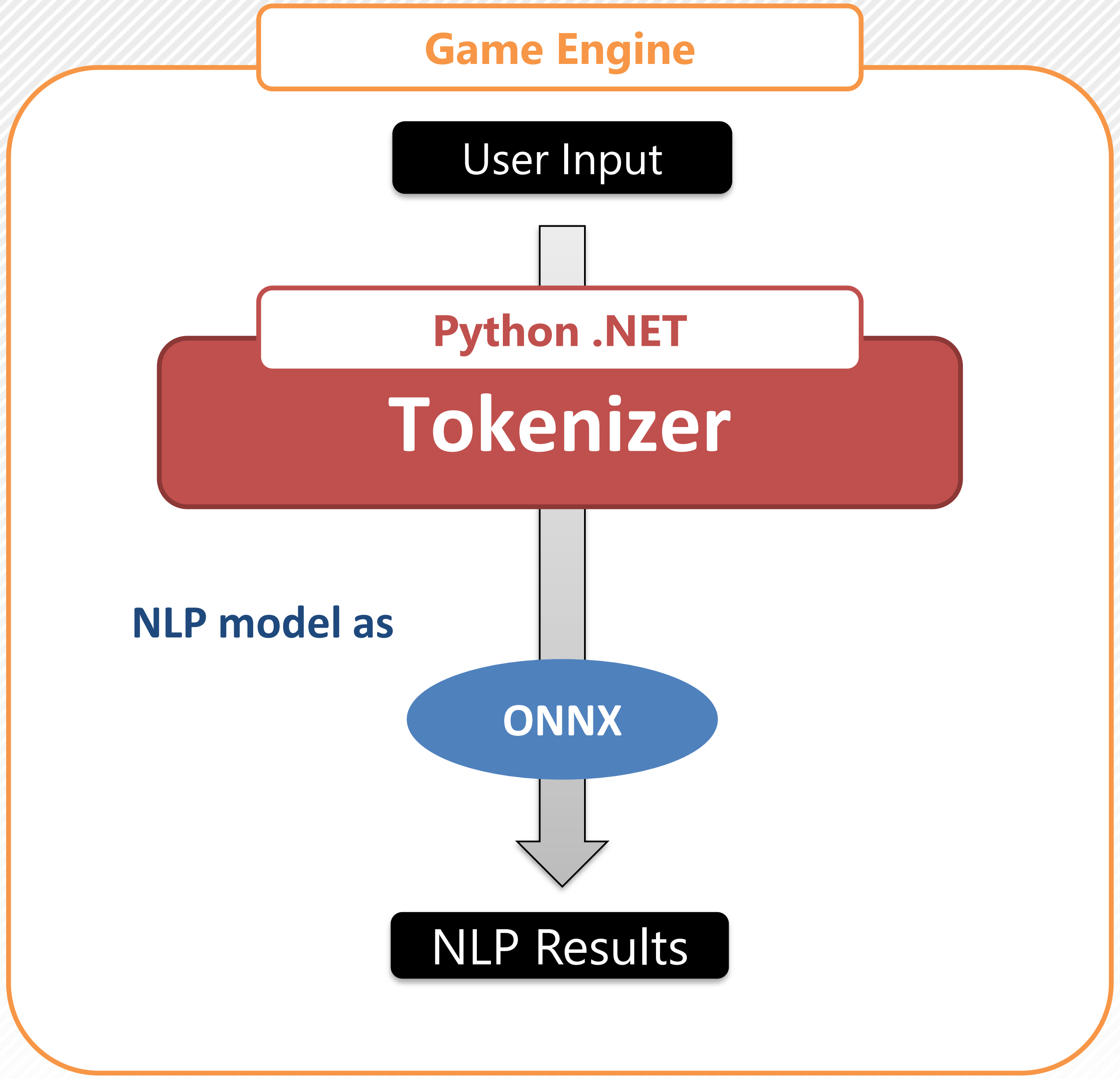
In a Game Script

- Use **Python .NET** to embed Python code in C#



```
public PyObject sampleTok;

public void LoadTokenizerSample()
{
    using (Py.GIL())
    {
        var tokPath = new PyString(tokPath);

        sampleTok = tokenizers.GetAttr("Tokenizer").InvokeMethod("from_file", pyTokPath);
    }
}


public OnnxInput Tokenize(string inputText)
{
    PyObject tokenized;

    OnnxInput tmpInput = new OnnxInput();

    using (Py.GIL())
    {
        var pystringInputText = new PyString(inputText);

        tokenized = sampleTok.InvokeMethod("encode", pystringInputText);

        tmpInput.InputIds = tokenized.GetAttr("ids").As<long[]>();
        tmpInput.AttentionMask = tokenized.GetAttr("attention_mask").As<long[]>();
    }

    return tmpInput;
}
```

One Application



Now,
you can convert your model and tokenizer and
have them work in your game!

How to Decide Models?

- **To be embedded into a game!**
  - Exportability to ONNX
  - Good performance for the required specifications

How to Decide Models?

- In general, the larger models with more parameters have higher performance
  - However, the specifications required to operate them are very high

How to Decide Models?

- If intended to be integrated into a game,
  it is necessary to consider the environment in which users will play

Our Choices of Models

As an Example

- **Natural Language Understanding (NLU)**
  - mSimCSE
    - Multi-lingual (no need to add a model when doing localization)
    - SimCSE can be trained in unsupervised manner

- **Natural Language Generation (NLG)**
  - DialoGPT (medium)
    - Well-known model and can be used as a baseline of ML-based generation
    - The size can be handled in CPU

Summary

Features in NLP Adventure

How can we evolve adventure games?

Free Input

Even when users use different words or phrases, the system can **understand users’ intent** and continue the scenario

Chit-chat

Even if a player input is unrelated to the story, the character responds to it

Experience

with NLP Adventure

- **“Discovery”**
  - Players discover their route by themselves, without given candidates
  - They don’t have to hear “I cannot understand it” again and again

- **“Sense of immersion”**
  - Input by their own words make players feel immersive into the game

Copyright

SQUARE ENIX, AI Division

Team

- AI Unit (2011-2021) → AI Division founded in 2022

- Now about 15 members (AI Experts)

  - AI Engineers, AI Researchers

- Research & Development:

  - covering a whole region of Game AI
  - development & academic research

Thank you!

- If you have any question, please don’t hesitate to contact:

Yusuke Mori
Artificial Intelligence Division
SQUARE ENIX CO., LTD.
moriyus@square-enix.com