Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/45/20241034

The application of Rasa framework in the interaction between players and game NPCs

Huangao Yao

Chongqing University, Chongqing, 401331, China

seyuki07s@gmail.com

Abstract. In the current era of thriving electronic games, some players have articulated a novel requirement for electronic games – the need for greater freedom and autonomy within the gaming experience. In traditional electronic games' interaction methods, there often exist issues of low engagement and insufficient immersion due to the limited narrative approach. At the same time, Rasa technology has made significant progress in the field of open-source dialogue systems. Rasa has the ability to understand and process multiple rounds of conversations, supports multilingual and multi-channel applications, and provides rich tools and components to enable developers to quickly build intelligent dialogue systems which is likely to be applied within the game industry. In the future, Rasa will continue to develop and provide people with a more intelligent and personalized conversation experience. This paper will explore a new application of Rasa platform for players to interact with non-player characters (NPCs) in electronic games which contributes to solving the issues.

Keywords: Rasa, Natural Language Processing, Video Games, Interaction.

1. Introduction

Natural Language Processing (NLP) is one of the theoretically advanced techniques, used for understanding human language and representing it [1], of which the most representative one is Rasa. Rasa is commonly used as a tool for building conversational systems, and it is also an open-source natural language understanding module [2]. It is composed of loosely coupled modules, combining multiple natural language processing and machine learning libraries, and providing a consistent application programming interface (API) [3]. Rasa technology covers various aspects of dialogue system development, including natural language understanding (NLU), dialogue management (DM), and natural language generation (NLG) and the operating structure of Rasa is as shown in figure 1.

The application of Rasa technology in the field of electronic games can bring more intelligent and interactive elements to the gaming experience. It learns to recognize the relationships between different words, derive precise meanings from the text, understand sentences in various situations, and consider the prior discourse context [4]. Besides, Rasa predicts a set of slot-labels and slot-values associated with different segments of the input rather than a sequence of slots for each input word [5].

The scalability and flexibility of Rasa enable game developers to customize as well as customize dialogue systems to adapt to different types of games and gameplay. They can integrate other game systems and algorithms, utilizing the powerful capabilities of Rasa to achieve more complex and diverse game dialogue interactions. The application of Rasa technology in the field of electronic games also

DOI: 10.54254/2755-2721/45/20241034

provides powerful tools and frameworks to create intelligent, personalized, and interactive game dialogue systems. Through voice or text interactions, dynamic plot generation, and intelligent dialogue management, Rasa can offer players a more immersive, personalized, and engaging gaming experience. This article aims to combine the Rasa robot framework with existing text games to create a more diverse dialogue system for players and enrich the gaming experience.

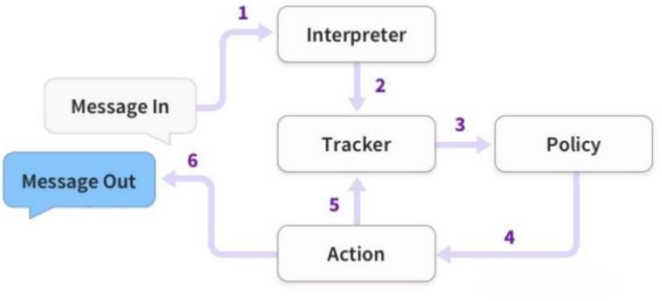

Figure 1. Rasa Core Operating Structure Diagram [6].

2. Overview of the Current Situation of Electronic Games

Video games are an ubiquitous part of almost all children's and adolescents' lives, with 97% of children and adolescents in the United States playing games for at least one hour per day [7]. In an article published in MIT's Technology Review, film scholar Henry Jenkins criticized Kroll for severely underestimating the potential of video games [8]. Electronic games require a sufficiently rich plot to provide attractiveness and appeal to the body like movies.

In the field of video game studies, narrative aspects of video games are often described in contrast to rule-based aspects [9]. In the highly prosperous online world, there are two mainstream storyline modes in electronic games [10]: Cinematic narrative, Alternative narrative.

However, both of the above narrative methods have the problem of low freedom, which has become a shackle that the gaming industry still can't break today.

Cinematic narrative is the use of movies to advance the plot. Players can't intervene or interact with the main plot and they can't determine the direction of the plot, just like watching a movie. The problem with it is that there is no freedom in the plot, and players play from a third perspective, resulting in insufficient participation and immersion.

Another option is alternative narrative, which mainly features the placement of different storytelling options in the game. Players can push the plot to different endings by selecting the chosen storytelling options set by the production team. This mode provides a certain degree of freedom, but there are still problems, such as fixed and non-adjustable alternatives, insufficient plot freedom, and the problem of plot fragmentation due to the independence and combination of alternatives.

In contrast, the main advantage of Rasa is its high degree of freedom, allowing for different answers even when saying the same words every time. Secondly, the dataset is easy to collect. For game companies, it is easy for screenwriters to write dataset samples that match the character's personality. Finally, there are multiple application scenarios where Rasa technology can be applied not only in the main storyline of games, but also in the development of dlc and game peripheral apps. The application of Rasa technology in the field of electronic games can provide powerful tools and frameworks for game developers to create intelligent, personalized, and interactive game dialogue systems. Through text interaction, dynamic storyline generation, and intelligent dialogue, Rasa can provide players with a more immersive, personalized, and interesting gaming experience.

3. Building Rasa Dialogue Robot by Using Existing Game Text

Building an NPC dialogue system in electronic games requires the use of existing corpora and modules such as NLU, DM, and NLG in the Rasa framework.

DOI: 10.54254/2755-2721/45/20241034

3.1. Obtaining corpus

This paper is based on the text of a visual novel game, which is used as a corpus to simulate NPC conversations after Data cleansing and labeling. To obtain the text of a visual novel game, there is a need of using the unpacking tool to unpack it (only for research, without any commercial use).

KIRIKIRI is a Japanese visual novel game engine widely used for producing and running visual novels and adventure games. This engine allows game developers to package game content and resources into specific package files, typically using extensions such as kirikiri, xp3, pimg, and Ks Due to the popularity and openness of the KIRIKIRI engine, many visual novels and adventure games have used it for development.

The unpacking software used in this article is KrkrExtract, which is a powerful tool for extracting and packaging xp3 files in engine games such as krkr2 and krkrz. The original address of the open-source GitHub code is marked in the reference file.

Figure 2. KrkrExtract and its dll file diagram.

As shown in Figure 2, using the KrkrExtract tool, it needs to drag the dll file of the sample game into the folder of the visual novel and click on the exe file to run it [11].

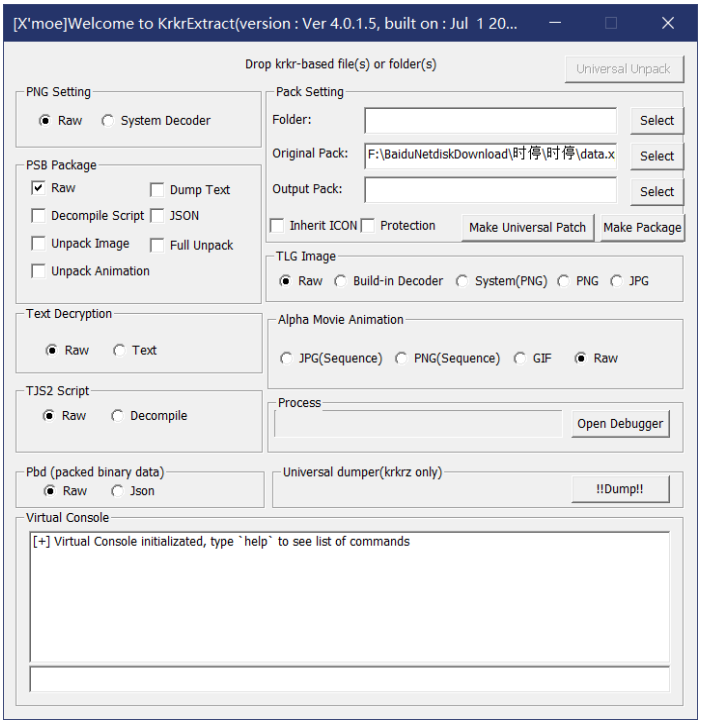

Figure 3. KrkrExtract Settings Interface ("时停" is the name of the game folder and can be replaced by any other word).

DOI: 10.54254/2755-2721/45/20241034

According to the settings in Figure 3, drag the xp3 file that needs to be unpacked into the “Original Pack” in the second column in the upper right corner, and the file will automatically unpack and generate the “KrkrExtract_Output” folder.

Entering the “KrkrExtract_Output” folder, there is a folder full of ks suffix files (folder names may vary depending on different game manufacturers). Here is the unpacked text file from the game, as shown in Figure 4.

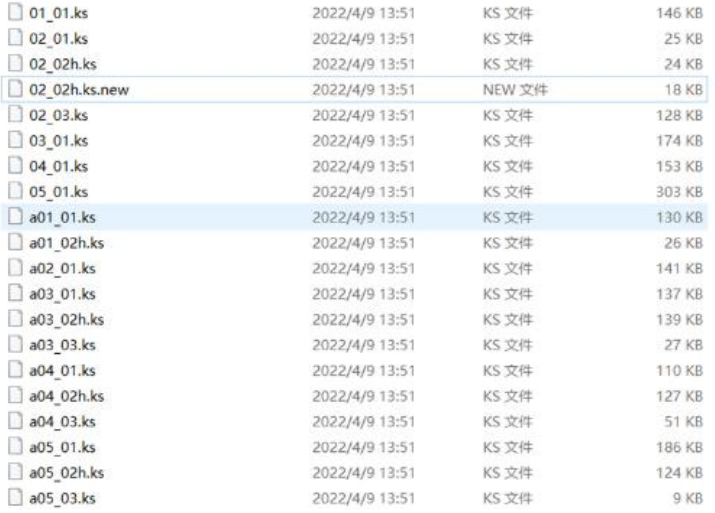

Figure 4. Unpacked Text File.

Since the files with the suffix ks can’t be edited, there is a need to convert them into txt files in batches to facilitate Data cleansing and reference later. This study uses a bat batch processing program for processing, as shown in Figure 5 after opening.

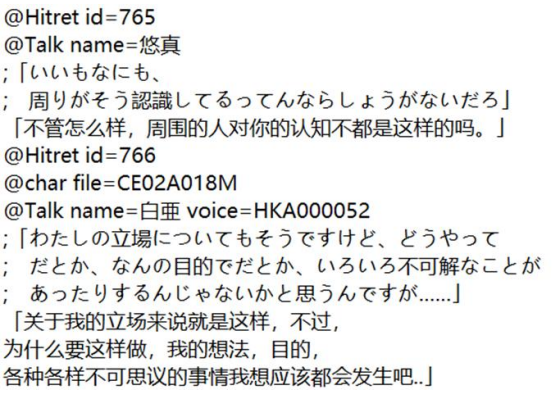

Figure 5. Sample txt file after batch processing (The sentences within the “「」” contain unpacked game text that can be replaced with English sentences.).

So far, the visual novel game is unpacked and its text is all obtained.

DOI: 10.54254/2755-2721/45/20241034

3.2. Overview of Detailed Steps for Establishing Rasa

3.2.1. Install Rasa (using the Windows 10 environment as an example) [12]

(1) Create a new Python project

It should be noted that in this step, it is necessary to ensure that Python 3.6 or Python 3.7 is installed in the Windows 10 environment. This paper is based on the Pycharm IDE development, version 2018.3.5 professional.

(2) Install Rasa

Open the Pycharm command terminal and enter the following command to install Rasa.

pip --default-timeout=500 install -U Rasa

After installation, enter the pip show Rasa command to view Rasa version information.

(3) Install MITIE and jieba

①Install MITIE

Download the MITIE source code and the Chinese word vector model. Here, it's needed to copy the model to the created Python project data directory, which will be used for training the NLU model later.

Then, install Visual Studio 2017, it's needed to check "Visual C++Tools for CMake" because compiling MITIE source code requires Cmake and the Visual C++environment. After installation, add C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\IDE\CommonExtensions\Microsoft\CMake\CMake\binto the environment variable and restart the computer to take effect.

Finally, enter the root directory of MITIE from the pychar command terminal and execute the following command:

python setup.py build

python setup.py install

MITIE can be installed.

②Install jieba

Enter the following command in the command terminal of pychar to install jieba pip install jieba

(4) Create Rasa configuration file

Executing the Rasa init command will automatically generate all necessary files for developing a Rasa project. With these files, the Rasa project can be run without any modifications. However, if automatic creation is not successful, there is a need to manually create Rasa configuration files in the Rasa root directory. The directory names and purposes of each configuration file are shown in Table 1.

Table 1. Rasa Configuration Files and Their Functions (from https:/Rasa.com/docs/).

__init__.py	an empty file that helps python find actions
actions.py	code for custom actions
config.yml	configuration of NLU and Core models
credentials.yml	details for connecting to other services
data/nlu.md	the NLU training data
data/stories.md	the stories
domain.yml	the assistant's domain
endpoints.yml	details for connecting to channels like fb messenger
models/<timestamp>.tar.gz	the initial model

3.2.2. Building NLU samples

Using annotated training data to train NLU models enables NPCs to accurately understand players' intentions and extract entity information. Suitable NLU pipelines and algorithms can be selected to train the model, such as Space, Jieba, or Mitie.

In this paper, the next step is to open the nlu.md file and build an NLU model for testing.

DOI: 10.54254/2755-2721/45/20241034

3.2.3. Building Core Samples

(1) Build the stories.md file data

The data stored in the stories.md file is the model of the conversation, which can also be referred to samples for training the Core model.

(2) Building domain.yml file data

domain.yml file is equivalent to the brain of an AI assistant, which records all the information in the system.

3.2.4. Train NLU and CORE models

(1) Configure the config.yml file

Configure the config.yml file to train NLU and Core models.

(2) Model training

Open the Pycharm command terminal and execute the following command. This command will train both NLU and Core models, as follows:

python -m Rasa train --config configs/config.yml --domain configs/domain.yml --data data/

The instruction parameters in the instruction terminal are explained in the official Rasa robot manual.

3.2.5. Configure Http and Action

(1) credentials.yml

credentials.yml is used to configure detailed authentication information for connecting to other services. When there is a need to access Rasa Server through HTTP, it's needed to configure rest in this file. The rest channel will provide a rest endpoint (Rasa Server) for sending messages to it, which will respond to requests and send back bots messages. We request that the URL of the Rasa Server be:

http://RasaServerIP:RasaServerPort/webhooks/rest/webhook

(2) action.py

When Rasa NLU recognizes the user's intention to input a Message, the Rasa Core dialogue management module will respond to it, and the module that completes this response is the action. Rasa Core supports three types of actions, which are default actions, utter actions and custom action.

3.2.6. Start Service

(1) Start Rasa Service

Enter the following command in the command terminal:

python -m Rasa run --port 5005 --endpoints configs/endpoints.yml --credentials configs/credentials.yml –debug

This service implements NLU and Core functions.

(2) Start action service

Enter the following command at the pychar command terminal:

Python -m Rasa run actions --port 5055 --actions actions –debug

(3) Create and start server.py

After configuration is completed, enter the following command to start:

python server.py

So far, the construction of the proposed Rasa robot service framework has been completed. The next step is to optimize the NPC like character of Rasa robots through visual novel game text.

4. Optimization of Rasa robot that simulating NPC

4.1. Define intention and entity

Firstly, the intentions and entities required for different NPC dialogue systems in the game should be defined, which represent the player's possible intentions or instructions, such as "greetings", "inquiries", etc. Entities represent information that NPCs may need to collect, such as name, status, etc. These intentions and entities will help NPCs understand player input and respond accordingly.

DOI: 10.54254/2755-2721/45/20241034

4.2. Collect and annotate training data

Based on the visual novel texts which have been collected in Section 3, training data related to NPC conversations can be collected, including player input and corresponding NPC responses. Annotate these conversation data to mark intentions and entities for training the NLU model. The training data annotation tool provided by Rasa can be used to simplify this process.

4.3. Define dialogue process

Design the dialogue process for the NPC dialogue system, including predefined stories, which are revised through the stories.md file in the Rasa robot root directory. The story describes the interaction process between players and NPCs, including player input, NPCs' responses, and expected next steps. By designing multiple stories, NPCs can learn and simulate different dialogue scenarios.

4.4. Training Dialogue Management (DM) Model

Use predefined stories and conversation data to train the DM model, so that NPC can respond according to the player's input. A rule driven conversation manager can be chosen or the reinforcement learning algorithm can be used to optimize the conversation strategy so as to provide more intelligent and flexible responses.

4.5. Design NLG module

Design an NLG module to generate NPC response text. There are two ways to use templates and dynamically generate text. The template is a predefined text mode, which contains some replaceable variables. The variables are filled according to the actual situation to generate the final response. Dynamic text generation uses machine learning methods to generate natural language text based on contextual and semantic information.

4.6. Integrate NPC dialogue system into the game

Integrate the trained NLU, DM, and NLG models into the NPC characters in the game. Ensure that NPCs can respond to player input in real-time and engage in intelligent dialogue and interaction. Integration can be achieved using the APIs and plugins provided by Rasa.

4.7. Testing and optimization

After integrating the NPC dialogue system, testing and optimizing it is a continuous process of improvement and expansion, which can continuously optimize NPC response and interaction methods based on player feedback and needs.

5. Synthesis and Conclusions

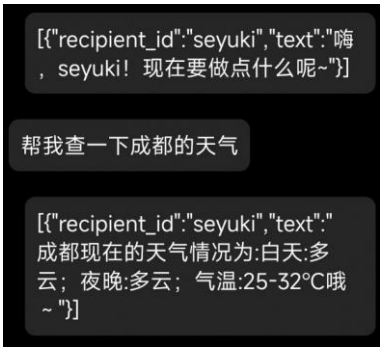

Figure 6. Sample communication text (This research uses Chinese as the default language which can be replaced by English).

DOI: 10.54254/2755-2721/45/20241034

In summary, the benefits of applying Rasa robot technology to the electronic gaming industry include providing automated customer support, delivering personalized gaming experiences, enhancing community interactions, offering game guidance and tutorials to players, as well as analysing player emotions and providing corresponding interactions. These advantages can enhance player experiences, elevate user satisfaction, and generate higher returns for the gaming industry.

However, this approach still has some shortcomings, including potential technological limitations and constraints. For instance, there may be limited understanding and answering capabilities for complex questions, a need for improved accuracy in emotion recognition and expression, and an inability to replicate the same authenticity and emotional resonance as human interactions. Additionally, robots lack human intuitive judgment and subjective experience, cannot fully cater to individualized needs, and require further enhancement in responding to real-time player feedback and complex situations. Therefore, within the electronic gaming industry, it's necessary to combine the strengths of artificial intelligence and human-crafted content to provide more comprehensive, in-depth, and personalized character design.

As artificial intelligence and natural language processing technologies continue to advance, Rasa robots are willing to be capable of more accurately understanding and answering complex questions, delivering more personalized gaming experiences, and establishing more authentic emotional interactions with players. Additionally, as conversational and voice interactions become more prevalent, Rasa robots can better adapt to various channels and devices, providing players with seamless gaming experiences. Looking ahead, Rasa robots are expected to combine with emerging technologies like virtual reality and augmented reality, further expanding the possibilities within the gaming industry and offering players even more immersive and personalized gaming experiences.

References

[1] Cambria E, White B 2014 Jumping NLP curves: a review of natural language processing research J. IEEE Computational Intelligence Magazine 9(2) p 48-57.
[2] Jiao A. An intelligent chatbot system based on entity extraction using Rasa NLU and neural network[C]//Journal of physics: conference series. IOP Publishing, 2020, 1487(1): 012014.
[3] Bocklisch T, Faulkner J, Pawlowski N and Nichol A 2017 Rasa: Open Source Language Understanding and Dialogue Management arXiv: Computation and Language
[4] Tembhekar S and Kanojiya M 2017 A Survey Paper on Approaches of Natural Language Processing (NLP) J. International Journal of Advance Research, Ideas and Innovations in Technology 3(3) p 1496-98.
[5] Desot T, Raimondo S, Mishakova A, Portet F and Vacher M 2018, September Towards a French Smart-Home Voice Command Corpus: Design and NLU Experiments C. text speech and dialog
[6] GitHub CI. (2023). https://rasa.com/docs/rasa/.
[7] Granic I, Lobel A, Engels R C M E. The benefits of playing video games[J]. American psychologist, 2014, 69(1): 66.
[8] Henry Jenkins, "Art Form for the Digital Age" (TechnologyReview, September/October 2000); see also Henry Jenkins,"Games, the New Lively Art," forthcoming in Jeffrey Goldstein(ed.), Handbook for Video Game Studies (Cambridge: MITPress).
[9] Koenitz H. Narrative in Video Games[J]. 2019.
[10] Ip B. Narrative structures in computer and video games: Part 1: Context, definitions, and initial findings[J]. Games and Culture, 2011, 6(2): 103-134.
[11] Hulotte. (2018) The inevitable destiny opened by an Atropos watch. http://hulotte.jp/product/atropos/
[12] Bocklisch T, Faulkner J, Pawlowski N, et al. Rasa: Open source language understanding and dialogue management[J]. arXiv preprint arXiv:1712.05181, 2017.