Python を用いた人工無能の制作

亀高　彩花

１．研究概要

コンピューターと人が自然な会話をさせることを目的として Python を用いて人工知能(無能)を制作した。

２．研究の具体的内容

(1)Python とは

図 1　Python のロゴ

Python はＣ言語をベースとして作られた言語で、文法を極力単純化してコードの可読性を高め、読みやすく、また書きやすくしてプログラマの作業性とコードの信頼性を高めることを重視してデザインされた、汎用の高水準言語である。さまざまなプログラムを分かりやすく、少ないコード行数で書けるといった特徴がある(図 2)。

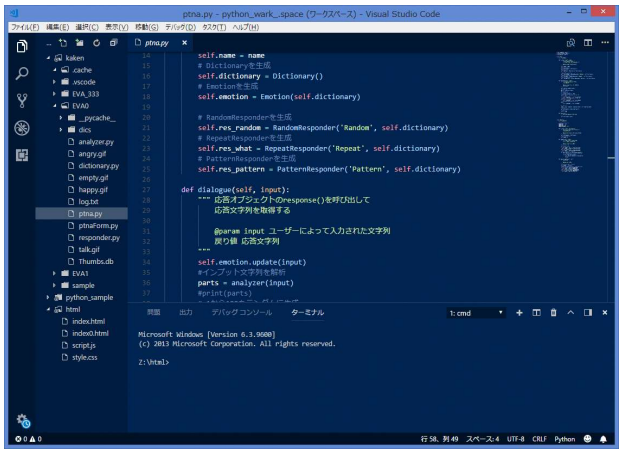

図 2　コーディング画面

(2)人工無能の定義

本研究では「人工知能」と人工無能を区別し、人工無脳の定義として、投げかけられた会話に対して最適な返答を返すことを目的として作られるものとし、「人工知能」については、人間の思考を完全再現し会話の意味、文脈を理解することを目的として作られるものとする。

製作する人工無能の名前は「ピティナ」とする。

(3)人工無能「ピティナ」の仕様

感情アルゴリズムを搭載し、投げかけられた言葉によって機嫌値やイラストを変化させる。三つの会話パターンを作成することで同じ言葉を入力した際にも返答にバリエーションを持たせる。

加えて、形態素解析を用いることで自ら返答パターンを生み出せるようにする。

イメージ画像

ピティナの喜怒哀楽に応じて入れ替わる画像。機嫌値の数値に対応して変わる。
嫌い、かわいいなどのユーザーの発言により変動する機嫌値に対応して変わる。

不機嫌(angry.gif)

上機嫌(happy.gif)

やや不機嫌(empty.gif)

平常時(talk.gif)

(4)操作方法

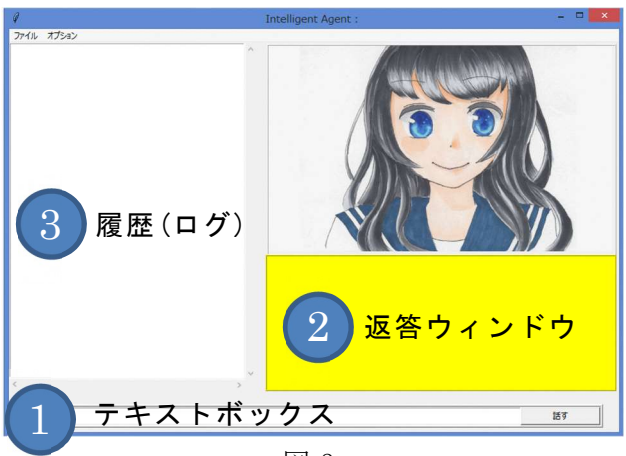

図 3

① テキストボックスに文章を入力してエンターキーを押し文章を確定させ、話すボタンをクリックする。

図 4

② 返答ウィンドウにピティナから返ってきた返事が表示される。

図 5

③ ログウィンドウに自分とピティナの会話した履歴が残る。

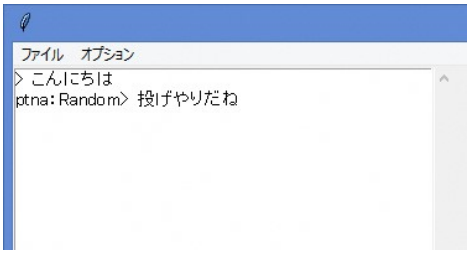

図 6

④ また、右上のバツ印をクリックすると確認画面が表示され、OK を押すとユーザーの発言した文字列がランダム辞書に格納される。

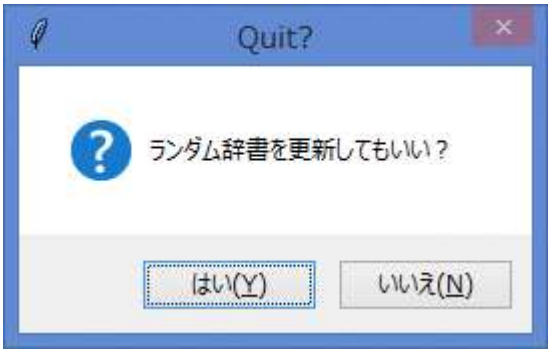

図 7

(5)形態素解析とは

Python のライブラリである「Janome」を利用して行うこととする。

形態素解析（けいたいそかいせき、Morphological Analysis）とは、文法的な情報の注記の無い自然言語のテキストデータ（文）から、対象言語の文法や、辞書と呼ばれる単語の品詞等の情報にもとづき、形態素（Morpheme，おおまかにいえば、言語で意味を持つ最小単位）の列に分割し、それぞれの形態素の品詞等を判別する作業である。
自然言語処理の分野における主要なテーマのひとつであり、機械翻訳やかな漢字変換など応用も多い（もちろん、かな漢字変換の場合は入力が通常の文と異なり全てひらがなであり、その先に続く文章もその時点では存在しないなどの理由で、内容は機械翻訳の場合とは異なったものになる)。

(6)メインプログラム

メインプログラム - ptna.py

class:Ptna

プログラムの本体クラス。基本的な文字列の操作はここで行う。関数 *dialoge* により、1 から 100 の乱数を発生させその数に応じ返答パターンを振り分ける。

表 1 – 乱数対応表

X≦60	PatternResponder
61≦X≦90	RandomResponder
それ以外	RepeatResponder

class:Emotion

ピティナ(人工無能の名前)の感情をつかさどるプログラム。

入力された文字列をパターン辞書と照合し、紐づけられた変動値を機嫌値に反映する方式をとる。

機嫌値の取る値は-15～15 の範囲内とし、いつまでも機嫌を損ねることの無いよう、ノーマルな応答を繰り返していくと一回につき 0.5 ずつ機嫌値を上昇させる処理を行う。機嫌値と機嫌の対応表を下の表 2 にて示す。

表 2

平常時(ー ー)	-5<=機嫌値<=5
やや不機嫌(゜Д゜)	-10<=機嫌値<=-5
不機嫌(# ゜Д゜)	-15<=機嫌値<=-10
上機嫌(*^^*)	5<=機嫌値<=15
書式のフォーマット	機嫌変動値##パターン[TAB]必要機嫌値##応答例 1|必要機嫌値##応答例 2| etc…

返答処理 - responder.py

class:Responder

応答クラスのスーパークラス。オーバーライドを前提としたプログラム。

class:RepertResponder

オウム返し(*RepertResponder*)を行うためのサブクラス。

return '{}ってなに？'.format(input)
{} にユーザーから取得した文字列を挿入して返答とする。

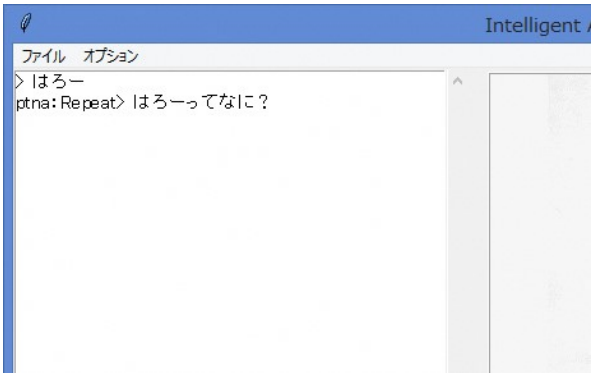

図 8

class:RandomResponder

ランダムな応答を返すためのサブクラス。ランダム辞書(リスト)から無作為に抽出した文字列を返答として返す。

図 9

class:PatternResponder

パターンに反応させるためのプログラム。インプットされた文字列にあらかじめ作成されたパターン辞書に適合するものがあればそれに則り返答を返す。
なお、パターンに適合しなければランダム辞書から返答する。

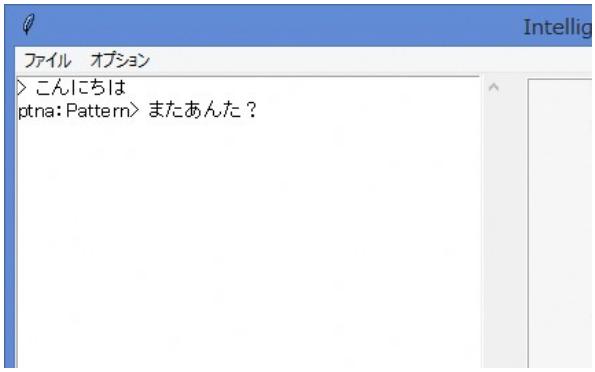

図 10

文字列処理 - dictionary.py

class:Dictionary

インプットされた文字列をランダム辞書に書き込んだり、パターン辞書と照合させたりするプログラム。ランダム辞書に書き込むことによりユーザーの発言を学習し、語彙を増やすことができる。

辞書ファイル - dics

pattern.txt

パターンマッチのための辞書。
(例)こんち(は|わ)$|こんにちはこんにちは|やほー|ちわす|ども|またあんた？

random.txt

ランダムな返答を返すための辞書。
(例) いい天気だね
今日は暑いね
おなかすいた

(7)会話アルゴリズム

会話パターンは“Repeat”“Pattern”“Random”の三つに分かれており、入力された文字列があらかじめ記述しておいたパターンに合致すれば Pattern 処理に、合致しなければその他の処理にランダムに割り振られるようになっている。

Random 処理ではランダム辞書と呼ばれる random.txt ファイルに書き込まれている文字列をランダムに出力する。

Repeat 処理では入力された文字列に“ってなに？”という文字列を連結して出力する。

３．研究のまとめ

この研究では Python の学習と自分の興味のあるプログラムを組むことを目的に人工無能の制作をしてきたが、周りを見ていてもチームでプログラムを組むことの大変さがわかった。すべて一人で組んでいったので進捗の共有やタスク管理などに追われることはなかったけれど、もしもチームでやっていたらと思うとその苦労は想像に難くない。大人数で制作している人は素直にすごいと感じた。

研究で学んだ内容は今後の趣味や学習に生かしていきたいと思う。

参考サイト
・Wikipedia　形態素解析
https://ja.wikipedia.org/wiki/形態素解析
参考書籍
・Python プログラミングパーフェクトマスター
https://www.shuwasystem.co.jp/products/7980html/4816.html