常熟理工学院

《人工智能原理与应用》实验指导及报告书

__2021___/___2022__学年　第__2_学期

姓　　名：___陆贤飞_____

学　　号：__093119112___

班　　级：__大数据 191__

指导教师：___赵露_____

计算机科学与工程学院

2022

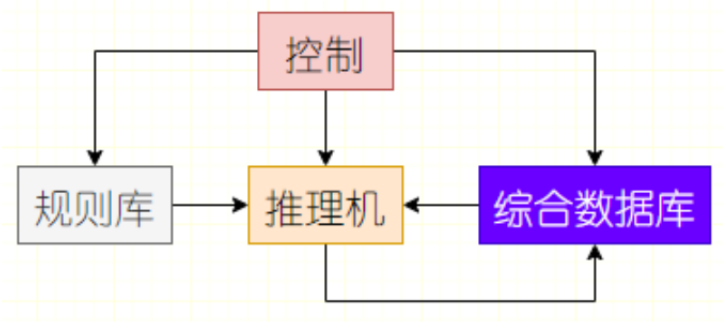
实验二　动物识别系统

一　实验目的

熟悉和掌握产生式系统的运行机制，掌握基于规则推理的基本方法。

二　实验原理

产生式系统用来描述若干个不同的以一个基本概念为基础的系统，这个基本概念就是产生式规则或产生式条件和操作对。在产生式系统中，论域的知识分为两部分：用事实表示静态知识；用产生式规则表示推理过程和行为。



规则库：用于描述相应领域内知识的产生式集合。
综合数据库：用于存放问题求解过程中各种当前信息的数据结构。
控制系统：由程序组成，负责整个产生式系统的运行，实现对问题的求解。

三　实验内容

1.　简述产生式系统的正向推理的过程

正向推理是一种从已知事实出发、正向使用推理规则的推理方式。

Note:

- 用户需要事先提供一组初始证据。并将其放入综合数据库
- 推理开始后，推理机根据综合数据库中的已有事实，到知识库中寻找当前可用的知识，形成一个当前可用的知识集
- 然后按照冲突消解策略，从该知识集中选择一条知识
- 使用选定的知识进行推理，并将新推出的事实加入综合数据库中，作为后面继续推理时可用的已知事实
- 如此重复直到求出所需要的解或者知识库中再无可用知识为止

2. 编程实现动物识别系统

2.1 根据以下 15 条规则，建立规则库

r_1: IF 该动物有毛发 THEN 该动物是哺乳动物
r_2: IF 该动物有奶 THEN 该动物是哺乳动物
r_3: IF 该动物有羽毛 THEN 该动物是鸟
r_4: IF 该动物会飞 AND 会下蛋 THEN 该动物是鸟
r_5: IF 该动物吃肉 THEN 该动物是食肉动物
r_6: IF 该动物有犬齿 AND 有爪 AND 眼盯前方 THEN 该动物是食肉动物
r_7: IF 该动物是哺乳动物 AND 有蹄 THEN 该动物是有蹄类动物
r_8: IF 该动物是哺乳动物 AND 是反刍动物 THEN 该动物是有蹄类动物

r_9: IF 该动物是哺乳动物 AND 是食肉动物 AND 是黄褐色 AND 身上有暗斑点 THEN 该动物是金钱豹
r_{10}: IF 该动物是哺乳动物 AND 是食肉动物 AND 是黄褐色 AND 身上有黑色条纹 THEN 该动物是虎
r_{11}: IF 该动物是有蹄类动物 AND 有长脖子 AND 有长腿 AND 身上有暗斑点 THEN 该动物是长颈鹿
r_{12}: IF 该动物有蹄类动物 AND 身上有黑色条纹 THEN 该动物是斑马
r_{13}: IF 该动物是鸟 AND 有长脖子 AND 有长腿 AND 不会飞 AND 有黑白二色 THEN 该动物是鸵鸟
r_{14}: IF 该动物是鸟 AND 会游泳 AND 不会飞 AND 有黑白二色 THEN 该动物是企鹅
r_{15}: IF 该动物是鸟 AND 善飞 THEN 该动物是信天翁

2.2 实现推理，打印推理过程

2.2.1 算法主要思路

设映射 $f : R \to C$，其中 R 是由上述 15 条规则的条件构成集合，C 是由上述 15 条规则的结论构成的集合。容易知道 f 是一个满射，则对任意 $y \in C$，都存在 $x \in R$，使得 $f(x) = y$，例如

$$f(\{\text{会飞}, \text{下蛋}\}) = \text{鸟}$$

于是可以把输入的事实看成一个集合 I，如果 $I \in R$，那么存在某个 $y \in C$，使得 $f(I) = y$，如果 $I \notin R$，如果能通过**添加元素规则**向 I 中添加有限个元素后，得到成 I'，使得 $I' = I_1 \cup \cdots \cup I_n$，其中 $I_i \in R \quad (\forall i = 1, 2, \cdots, n)$，则在这种**添加元素规则**下，集合 I 可以推出的结论为 $f(I_n)$，否则集合 I 不能推出任何结论。下面定义**添加元素规则**：

定义 2.2.1 如果有 $I \notin R$，$I_1, I_2, \cdots I_r \in R$，且 $I_1 \subset I$，那么就令

$$I_{r+1} = \left(I - \sum_{i=1}^{r} I_i\right) \bigcup_{i=1}^{r} \{f(I_i)\}$$

若 $I_{r+1} \in R$，则记。

$$I' = \bigcup_{i=1}^{r+1} I_i$$

举例说明上述思想的正确性，令

$$I = \{\text{毛发}, \text{吃肉}, \text{黄褐色}, \text{黑色条纹}\}$$

显然 $I \notin R$，故用添加元素规则添加元素，首先有 $I_1 = \{\text{毛发}\}$，$I_2 = \{\text{吃肉}\}$，且 $I_1, I_2 \subset I$，$I_1, I_2 \in R$，则有

$$I_3 = I - I_1 - I_2 \cup \{f(I_1), f(I_2)\} = \{\text{哺乳动物}, \text{食肉动物}, \text{黄褐色}, \text{黑色条纹}\} \in R$$

此时 $I' = I_1 \cup I_2 \cup I_3$，于是 I 可以推出的结论为 $f(I_3) =$ 虎

用 *Python* 中的集合数据结构容易实现，代码如下（其余代码均在附录）：

```
def deduce(self, keywords):
    """
    description: 进行推理
    param: keywords 关键词集合
    Returns: None
    """

    # 用于存放推理各步结论
    self.conclusion =  list()
    while  len(keywords) != 0:
        flag = 0
        for rule in self.rulebase:
            flag += 1
            # 判断集合R中是否有元素是输入事实的子集
            if rule.condition.issubset(keywords):
                # 减去该子集
                keywords = keywords - rule.condition
                # 添加子集的结论
                self.conclusion.append(rule)
                if  len(keywords) != 0:
                    keywords.add(rule.conclusion)
                    flag = 0
                    continue
                else:
                    break
        if flag ==  len(self.rulebase) and  len(keywords) != 0:
            self.conclusion =  list()
            break
```

读入规则集

```
def initialize_rulebase(self):
    """
    description: 从本地txt文件中读入规则
    param: None
    Returns: None
    """

    with  open( file=self.path, mode='r', encoding='utf_8') as  file:
        rules =  file.read().split("\n")
        for rule in rules:
            if  len(rule) != 0:
                self.rulebase.append(Rule(rule.split(" ")))
```

2.2.2 推理系统演示

图 1: 演示推理

图 2: 添加规则

图 3: 对添加的规则进行测试

提交至 *github* 链接

https://github.com/cgxf2021/AnimalSystem

附录

Listing 1: **animal_system.py**

```python
from email import header


class Rule( object):
    """
    description: 存储规则的数据结构
    """

    def __init__(self, rule) -> None:
        self.condition =  set(rule[0: -1])
        self.conclusion = rule[-1]


    def toString(self):
        return  str(self.condition) + '  ->  ' +  str(self.conclusion) + '\n'


```

```
18
19  class AnimalSystem( object):
20      """
21      description: 动物识别系统类
22      """
23
24      def __init__(self) -> None:
25          self.rulebase =  list()
26          self.conclusion =  list()
27          self.path = "./data/rulebase.txt"
28
29
30      def initialize_rulebase(self):
31          """
32          description: 从本地txt文件中读入规则
33          param: None
34          Returns: None
35          """
36
37          with  open( file=self.path, mode='r', encoding='utf_8') as  file:
38              rules =  file.read().split("\n")
39              for rule in rules:
40                  if  len(rule) != 0:
41                      self.rulebase.append(Rule(rule.split(" ")))
42
43
44      def add_rule(self, rule):
45          """
46          description: 添加新规则
47          param: rule 新的规则字符串类型
48          Returns: None
49          """
50          with  open( file=self.path, mode='a', encoding='utf_8') as  file:
51               file.writelines(rule + "\n")
52          self.rulebase.append(Rule(rule.split(" ")))
53
54
55      def deduce(self, keywords):
56          """
57          description: 进行推理
58          param: keywords 关键词集合
59          Returns: None
60          """
61
62          # 用于存放推理各步结论
63          self.conclusion =  list()
64          while  len(keywords) != 0:
```

```
            flag = 0
            for rule in self.rulebase:
                flag += 1
                # 判断集合R中是否有元素是输入事实的子集
                if rule.condition.issubset(keywords):
                    # 减去该子集
                    keywords = keywords - rule.condition
                    # 添加子集的结论
                    self.conclusion.append(rule)
                    if  len(keywords) != 0:
                        keywords.add(rule.conclusion)
                        flag = 0
                        continue
                    else:
                        break
            if flag ==  len(self.rulebase) and  len(keywords) != 0:
                self.conclusion =  list()
                break


    def output(self):
        """
        description: 输出结果列表中的结果
        param: None
        Returns: None
        """

        if  len(self.conclusion) != 0:
            for con in self.conclusion:
                print(con.condition, con.conclusion, sep=' -> ')
        else:
            print("该知识超出了认识范畴")
```

Listing 2: **form.py**

```
import tkinter as tk
from tkinter import ttk
from tkinter import scrolledtext
from animal_system import AnimalSystem
from animal_system import Rule


class Form( object):
    def __init__(self) -> None:

        self.win = tk.Tk()
        self.animal = AnimalSystem()
        self.rulebase =  str()
```

```
14          self.conclusions = tk.StringVar()
15
16          def initialize_rulebase():
17              # 初始化规则库
18              with  open( file="./data/rulebase.txt", mode='r', encoding='utf_8') as  file:
19                  rule =  file.read()
20              self.rulebase = rule
21              self.rulebase_text.insert(tk.INSERT, self.rulebase)
22
23          self.rulebase_label = ttk.Label(self.win, text="规则库", compound="center")
24          self.input_label = ttk.Label(self.win, text="输入事实", compound="center")
25          self.rulebase_text = scrolledtext.ScrolledText(self.win, width=40, height=25, wrap=tk.WORD)
26          self.input_text = scrolledtext.ScrolledText(self.win, width=50, height=10, wrap=tk.WORD)
27          self.deduce_button = ttk.Button(self.win, text="强行推理", width=8)
28          self.deduce_process_label = ttk.Label(self.win, text="推理过程", compound="center")
29          self.deduce_process_text = scrolledtext.ScrolledText(self.win, width=50, height=10, wrap=tk.
                WORD)
30          self.add_new_rule_text = ttk.Entry(self.win, width=40)
31          self.animal_label = ttk.Label(self.win, text="是啥动物？")
32          self.add_new_rule_button = ttk.Button(self.win, text="添加新规则", width=10)
33          self.animal_text = ttk.Entry(self.win, width=40)
34          initialize_rulebase()
35          self.animal.initialize_rulebase()
36
37
38      def create_form(self):
39          """
40          description: 创建界面
41          param: None
42          Returns: None
43          """
44
45          # 设置主窗口
46          self.win.title("Animal System")
47
48          # 设置各部件位置
49          self.rulebase_label.grid(column=0, row=0, padx=10, pady=5)
50          self.input_label.grid(column=1, row=0, padx=10, pady=5)
51          self.rulebase_text.grid(column=0, row=1, rowspan=4, padx=10)
52          self.rulebase_text.configure(state='disabled')
53          self.input_text.grid(column=1, row=1, padx=10)
54          self.deduce_button.grid(column=1, row=2, padx=20, pady=5, sticky=tk.E)
55          self.deduce_process_label.grid(column=1, row=3, padx=10, sticky=tk.W)
56          self.deduce_process_text.grid(column=1, row=4, padx=10)
57          self.add_new_rule_text.grid(column=0, padx=10, pady=8, sticky=tk.W)
58          self.animal_label.grid(column=1, row=5, padx=10, pady=5, sticky=tk.W)
59          self.add_new_rule_button.grid(column=0, row=6, padx=10, pady=10)
```

```
60          self.animal_text.grid(column=1, row=6, padx=10, sticky=tk.W)
61          self.animal_text.configure(state='disabled')
62
63
64          def click_deduce_button():
65              """
66              description: 设置按钮回调函数
67              """
68
69              # 记录最终推理结果的字符串
70              conclusions =  str()
71              # 从输入事实文本框读入字符串
72              feature_str = self.input_text.get('1.0', 'end-1c')
73              # 处理字符串
74              if  len(feature_str) != 0:
75                  feature = feature_str.split('\n')
76                  for item in feature[0: -1]:
77                      keywords =  set(item.split(' '))
78                      # 调用animalsystem类中deduce方法进行推理
79                      self.animal.deduce(keywords=keywords)
80                      if  len(self.animal.conclusion) != 0:
81                          for con in self.animal.conclusion:
82                              self.deduce_process_text.insert(tk.INSERT, con.toString())
83                          self.deduce_process_text.insert(tk.INSERT, '\n')
84                          conclusions +=  str(self.animal.conclusion[-1].conclusion) + ";"
85                      else:
86                          self.deduce_process_text.insert(tk.INSERT, "该知识超出了认识范畴\n\n")
87                          conclusions += '?;'
88                  self.conclusions. set(conclusions)
89                  self.animal_text.configure(textvariable=self.conclusions)
90              else:
91                  # 读入字符串为空
92                  self.conclusions. set('???')
93                  self.deduce_process_text.insert(tk.INSERT, "输入事实为空\n\n")
94                  self.animal_text.configure(textvariable=self.conclusions)
95
96
97          def click_add_new_rule_button():
98              """
99              description: 添加新规则回调函数
100             """
101
102             rule_str = self.add_new_rule_text.get()
103             rule = rule_str.split(' ')
104             if  len(rule) >= 2:
105                 rule = Rule(rule=rule)
106                 self.animal.add_rule(rule=rule_str)
```

```
107                 self.rulebase += rule_str + '\n'
108                 self.rulebase_text.configure(state='normal')
109                 self.rulebase_text.insert(tk.INSERT, rule_str + '\n')
110                 self.rulebase_text.configure(state='disabled')
111             self.add_new_rule_text.configure(textvariable=tk.StringVar(value=""))
112             self.animal.initialize_rulebase()
113
114
115         # 设置推理按钮回调函数
116         self.deduce_button.configure(command=click_deduce_button)
117
118         # 设置添加规则按钮回调函数
119         self.add_new_rule_button.configure(command=click_add_new_rule_button)
120
121         self.win.mainloop()
122
123
124 if __name__ == "__main__":
125     form = Form()
126     form.create_form()
```