Coding a Disease diagnosis system using First Order Logic (FOL)

Problem introduction

- Let's consider a simple diagnostic system using First-Order Logic (FOL) to diagnose three diseases: Influenza (flu), Common Cold, and Pneumonia. We'll define some common symptoms associated with each disease and represent the diagnostic rules using FOL.

1. Influenza (Flu): Symptoms (fever, cough, body aches)
2. Common cold: Symptoms (runny nose, sneezing, sore throat)
3. Pneumonia: Symptoms (fever, cough, shortness of breath)

Representation of the problem using FOL

1. Define predicates:
   1. Disease(x): Represents that x is a disease.
   2. Symptom(x, y): Represents that person x exhibits symptom y.
2. Define the symptoms for each disease:
   1. Disease(Influenza)
      1. Symptom(x, Fever) ∧ Symptom(x, Cough) ∧ Symptom(x, BodyAches)
   2. Disease(CommonCold)
      1. Symptom(x, RunnyNose) ∧ Symptom(x, Sneezing) ∧ Symptom(x, SoreThroat)
   3. Disease(Pneumonia)
      1. Symptom(x, Fever) ∧ Symptom(x, Cough) ∧ Symptom(x, ShortnessOfBreath)
3. Diagnostic rules:
   1. If a person exhibits the symptoms of a particular disease, we can conclude that the person has that disease.
   2. For example:
      1. Disease(x) ∧ Symptom(y, Fever) ∧ Symptom(y, Cough) ⇒ Disease(Influenza)

Coding

- In the previous section, we had two predicates; Disease and Symptom
- We are going to create classes for each of these predicates and a function, diagnose, that takes as input a list of diseases and a list of symptoms and figures out whether the diseases would be diagnosed based on the symptoms.
- Create a new folder called ‘chapter_3’. In this folder create a file called fol.py

Coding - predicates

- Paste the following classes into the fol.py file created in the previous section.

```
class Disease:
    def __init__(self, name, symptoms):
        self.name = name
        self.symptoms = symptoms

class Symptom:
    def __init__(self, name):
        self.name = name
```

Coding – diagnose function

- Before we go about coding, let’s understand the logic of diagnosis.
- A particular disease can be diagnosed if and only if all of its symptoms are present in the list of observed symptoms of a patient.
- Let’s say we have a patient with fever, cough and body aches, we can confidently say, based on our knowledge, that they have the flu.
- Take note that this diagnosis can result in more than one disease. Say our patient has fever, cough, body aches and shortness of breath.
- Based on our knowledge, we will say the patient has both the flu and pneumonia.
- Just fever or cough cannot result in an accurate diagnosis.

Coding – diagnose function

- Ok, so following our description on the previous slide, we are going to create a function called diagnose.
- This function will take in a list of diseases and a list of symptoms.
- For each disease in our list of diseases, it checks if all of the disease’s symptoms are in the list of symptoms passed as parameters.
  - If the above check is true, the disease is added to a list of diagnosed diseases.
- This list of diagnosed diseases is then returned by the function

Coding – diagnose function

- Paste the following code into the fol.py created earlier.

```
def diagnose(diseases, symptoms):
    diagnosed_diseases = []

    for disease in diseases:
        if all(symptom in symptoms for symptom in disease.symptoms):
            diagnosed_diseases.append(disease)

    return diagnosed_diseases
```

Coding – diagnose function

- Paste the following code into the fol.py file.

```
# symptoms
fever, cough, body_aches = Symptom(“fever”), Symptom(“cough”), Symptom(“body aches”)
runny_nose, sneezing = Symptom(“runny nose”), Symptom(“Sneezing”)
short_breath, sore_throat =  Symptom(“short breath”), Symptom(“sore throat”)

#diseases
influenza = Disease(“Influenza”, [fever, cough, body_aches])
common_cold = Disease(“Common cold”, [runny_nose, sneezing, sore_throat])
pneumonia = Disease(“Pneumonia”, [fever, cough, short_breath])

diseases = [influenza, common_cold, pneumonia]
```

Coding – diagnose function

- Paste the following code into fol.py

```
if __name__ == “__main__”:
    patient_symptoms = [fever, cough]

    diagnosed_diseases = diagnose(diseases, patient_symptoms)

    if diagnosed_diseases:
        print(“The patient may have the following diseases”)
        for disease in diagnosed_diseases:
            print(disease.name)
    else:
        print(“The patient’s symptoms do not match any known diseases”)
```

Coding – running the script

- Try running the script. It should print
- Try adding body_aches to the list of patient symptoms ([fever, cough, body_aches]), it should display ‘Influenza’.
- Try adding short_breath to the list of patient symptoms too ([fever, cough, body_aches, short_breath]), it should display Influenza and Pneumonia.

Introducing forward and backward chaining

- Our current diagnostic system is very limited and cannot really diagnose even for incomplete symptoms (e.g. if you change the patient symptoms to [runny_nose, sneezing] it won’t still diagnose common_cold, whereas that is the only possible diagnosis.
- We will create a class for our diagnostic system that will use forward and backward chaining to diagnose diseases based on symptoms.

Coding – forward chaining

- Paste the following code into the fol.py file, before the `if __name__ == '__main__'` block

```
class DiagnosticSystem:
    def __init__(self, diseases):
        self.diseases = diseases

    def forward_chaining(self, symptoms):
        diagnoses = []
        for disease in self.diseases:
            symptoms_list = disease.symptoms

            if all(symptom in symptoms_list for symptom in symptoms):
                diagnoses.append(disease)
        return diagnoses
```

Coding – backward chaining

- Paste the following code in the DiagnosticSystem class created above

```
def backward_chaining(self, disease, symptoms):
    if disease in self.diseases:
        if all(symptom in disease.symptoms for symptom in symptoms):
            return True
    return False

def get_diseases_string(self, diseases):
    return “,  ”.join(disease.name for disease in diseases)
```

Coding – testing the new system

- Paste the following code into the `if __name__ == ‘__main__’`

```
diagnostic_system = DiagnosticSystem(diseases=diseases)

# Forward chaining test
forward_diagnoses = diagnostic_system.forward_chaining([runny_nose, sneezing])
print(“Possible diagnoses: ”, diagnostic_system.get_diseases_string(forward_diagnoses))

# Backward Chaining Test
print(“Backward chaining test”)
backward_diagnosis = diagnostic_system.backward_chaining(influenza, [fever])
print(“is the patient suffering from influenza? ”, backward_diagnosis)
```