Imagery-Based Architecture

MD Asif Hasan

10/19/2017

Overview

The goal of this assignment is to develop a imagery-based system architecture that can be used for spatial reasoning. Imagery-based system is an analogical cognitive system that represents information in same way as received during perception. This implementation does not use real images as input, but tries to mimic image perception with high level image object. It receives images of objects as inputs and supports querying on spatial relationship between objects.

Development Platform: Python

Inputs

There are basically two types of inputs, Image and Query. Image is represented as a set of Objects. Each object also stores spatial relationship between other objects in that image. A helper function is implemented as part of the assignment which converts textual inputs into image data structure. For example, the code to create an input image is: *create_input_image (Input ("right", "plate", "fork"))* . The resulting image contains two objects, one for “plate” and another for “fork”. The object “plate” contains a relation “right” with “fork” and “fork” contains a relation “left” with “plate”. It is assumed that, a real world picture can be processed and then represented into this Image data structure and then fed into the system as input. A different helper function will need to be implemented for that case.

For querying spatial relationship between two objects the architecture provides a function that takes as input the text labels of the objects. For example: query("fork", "knife").

Outputs

When the imagery-based spatial reasoning system is queried on spatial relationship between two objects, it prints the relationships that holds true between the objects on console as well as return the relations as output of the query function as a dictionary which maps the relation label as keys and object labels as values.

System Description

The system is designed to support spatial reasoning of arbitrary spatial relationships. It takes in images as inputs which contains information on the objects it contains as well as the spatial relationships between them. They system uses a short term memory (STM) to store the images. STM stores the images as such that the set of images that are currently in memory and contains information on a specific object can be retrieved without searching through all the images in memory which supports faster reasoning.

When initializing the system a set of spatial relation types must be provided. The system uses these relation types to support querying. These types can be thought of as limitation of the implemented system and as they are provided from outside of the system makes it generic to any type of relations.

For querying spatial relation between two objects, the system uses searching through images. The algorithm is given below:

Query Algorithm

Function: Query (a, b)
Step 1:
Find all the relations that hold true between object a and b using QueryRelation(a, b, target_relation)
Step 2:
Print the relations that holds true between a and b
If none holds true, then print “I don’t know”
Return the relations that holds true between a and b
Function QueryRelation(a, b, target_relation)
Step 1:
Frontier = list containing object b
Explored = empty list
Step 2:
Set CO = pop the first element of Frontier
Step 3:
Test if CO == a
If yes, return True // target relation holds true between a and b
Step 4:
Find all the images containing object CO
Step 5:
If any of these images contains an object which is related by the target relation with CO, enlist that object in the frontier.
Step 6:
Enlist the processed images in the explored list
Step 7:
Repeat to Step 2 if Frontier is not empty
Step 8:
Return False // target relation does not hold true between a and b

Executing the Program

The program requires python 3.6 to run. To run the program the following command is to be executed from the command prompt.

python ImageryArchitecture.py <test_no>

<test_no> may be any number from 1 to 6.

Example Problem:

Let's walk through an example problem given below and see how the system goes about solving it.

Q1: The fork is to the left of the plate. The plate is to the left of the knife. Where is the fork, in relation to the knife?

A: left = true, which means fork is to the left of knife.

The system is first initialized with the following:

```
imageArch = ImageryArch(["left", "right", "above", "bottom"])
```

And, then, the images are fed into the system. To create the images from textual inputs "create_input_image" function is used.

```
imageArch.stm.addImage(create_input_image(Input("left", "fork", "plate")))
imageArch.stm.addImage(create_input_image(Input("left", "plate", "knife")))
```

After these steps, the short term memory will contain two images each containing two objects. The state of the images is shown below:

Image 1:
- Object 1: fork
  - relation : < right, plate>
- Object 2: plate
  - relation : < left, fork>

Image 2:
- Object 1: plate
  - relation : < right, knife>
- Object 2: knife
  - relation : < left, plate>

The short term memory stores the images in a dictionary with object labels as keys. The state of that dictionary at this point is, Dictionary: <fork, [Image 1]>, <plate, [Image 1, Image 2]>, <knife, [Image 2]>.

Now, the query function is called as: *imageArch.query("fork", "knife")* and it queries for each of the relation type separately.

At first, it queries between fork and knife for relation type "left'. The frontier is initialized to contain the second query object "knife" and starts exploring with "knife". The algorithm will stop when the exploring object is same as the first query object, in this case the "fork".

So, the algorithm is exploring "knife" and there is only a single image (Image 2) containing information on "knife" in the STM. The "knife" object in that image contains a relation of type "left" and that relation holds with the object "plate". So, the frontier is extended to include "plate". At this point, the frontier contains only a single object, which is the newly added object "plate".

At next iteration, the algorithm explores “plate” and both of the images contain information on “plate”. But, Image 2 is already explored, so the algorithm checks the relations contained in Image 1. According to that, “plate” has has a relation of type “left” with “fork”. So, the “fork” is added into the frontier list.

At the next iteration, the object “fork” is explored and the goal test which tests if the exploring object is same as the first query object, results to be true. Thus it returns true as the result of query of whether the relation “left” holds true between “fork” and “knife”.

At next, the algorithm queries for the relation type “right” between “fork” and “knife”. It starts with exploring knife. The knife object in the only image (Image 2) containing information of it does not hold any relation of type “right”. So, the frontier is not extended any further. At the end of the iteration, as a result of the frontier being empty, the algorithm returns false as the result of query of whether the relation “right” holds true between “fork” and “knife”.

Similarly the algorithm finds out that the rest of the relations (above and bottom) also do not hold true between “fork” and “knife”.

Finally the algorithm prints the following as the answer:
Spatial relation between fork and knife :
left = true

Results

For the queries listed in the assignment instruction, the results are shown below along with the query.

Q1: The fork is to the left of the plate. The plate is to the left of the knife. Where is the fork, in relation to the knife?

A: left = true, which means fork is to the left of knife.

Q2: The fork is to the left of the plate. The plate is above the napkin. Where is the fork, in relation to the napkin?

A: I don't know.

Q3: The fork is to the left of the plate. The spoon is to the left of the plate. Where is the fork, in relation to the spoon?

A: I don't know.

Q4: The fork is to the left of the plate. The spoon is to the left of the fork. The knife is to the left of the spoon. The pizza is to the left of the knife. The cat is to the left of the pizza. Where is the plate, in relation to the cat?

A: right = true, which means the plate is to the right of cat.

Additional Problems

Query 5 tests whether the systems performs correctly for reasoning complementary relations between objects.

Q1: The plate is to the right of the fork. The knife is to the right of the plate.. Where is the fork, in relation to the knife?

A: left = true, which means fork is to the left of knife.

In this case, the inputs were given between the objects with the relation type “right” and the specific query needed the system to provide information with the complementary relation type. The system performed correctly on that.

Query 6 is another special case testing which mixes complementary relation types in the input and tests whether the system performs correctly on that scenario.

Q1: The plate is to the right of the fork. The knife is to the right of the plate. The pizza is to the left of the fork. Where is the pizza, in relation to the knife?

A: left = true, which means pizza is to the left of knife.

For this case, the inputs were mixed with relations of type “right” and “left” and the query was given that needed the system to output relation type of “left”.

Discussion

The current implementation is very limited. It supports images as inputs that may contain information on two objects only. It will be interesting to see how images with more objects may affect the design. Furthermore, the data structure set for the image works well for this assignment. But, how it will perform or need be extended to support real world pictures is another point of interest. Overall the algorithm seem to be quite general to reason about basic spatial relationships. The burden of setting spatial relations between two objects is cleverly disassociated from the querying algorithm and is given to the Image creation function which is not part of the architecture. But, for complex relations between objects how the overall system will perform is not really realized.

The kind of imagery-based processes that are listed in Thagard (2005), such as transform, zoom in-out, etc, are never used in this architecture. How those kind of actions may come into play for solving reasoning problems is not realized as part of the experience implementing this architecture.

One major implication of this implementation of imagery-based reasoning is worth noting here. That is, though the system uses representation of images to reason, it still uses rules to perform query. For example, the algorithm uses a rule: if there is a path between two objects where each link of the path is of a specific relation type then the two objects are related with that relation type. For more complex reasoning, more complex rules will be necessary.

This system is not much different than the production system architecture developed for previous assignment. Though, it uses a different kind of structure to hold the facts about the world, it is analogous to the data structure (Conditions) used in production system. If the imagery-based architecture implemented here used proper image representation which is pixel based the difference probably would have been much clearer. Even in that case, there must have been some processes that would analyze the images to find out relations between the

objects and would represent them in symbolic structure. But, it is not unimaginable that, systems may exist which does not use any symbolic representation of spatial relations between objects at all.

They query mechanism is different between the production system and the imagery-based architecture. But, both uses search and rules. In both systems, the search produces new facts about the world and uses rules to test goals and explore new frontiers.

As a conclusion remarks, it is not very clear how imagery-based architecture is very different from production system architecture. It is different in many manner but not in terms of the core principles used to solve reasoning problems. So according to this implementation, the imagery-based architecture in human may actually use symbol-based processing architectures in human brain. It really seemed a decent idea to mix the production system architecture with the imagery based architecture where the visual perceptions will be processed by the imagery-based system into conditions that the production system then can use to reason.

References

Thagard, P. (2005). Mind: Introduction to cognitive science (2nd ed.). The Mit Press