ANSWERS

8.7 Models as text

BugBotLab

Learning · Robot club · about 15 min

What this lesson is about

Saving and loading a model with JSON; models trained elsewhere.

- JSON
- A model from somewhere else
- Using it

Questions

6 marks in all

1. What does this program print? [1 mark]

```
import json
sample = ([16, 16], "wall-ahead")
back = json.loads(json.dumps(sample))
print(back)
print(back == sample)
```

Answer:

```
[[16, 16], 'wall-ahead']
False
```

JSON has no tuples, so the pair comes back as a list, and a list is not equal to a tuple. That is why the models store samples as small dictionaries.

2. What do `json.dumps` and `json.loads` do? [1 mark]

- ☐ **A** `dumps` turns data into text; `loads` turns text back into data
- ☐ **B** `dumps` deletes a model; `loads` restores it
- ☐ **C** `dumps` saves to a file; `loads` downloads from the internet
- ☐ **D** Both turn text into data

Answer: A. Text is what you can print, copy, paste into a file or send to a friend.

3. What does this program print? [1 mark]

```
import json
MODEL = '{"kind": "nearest-neighbour", "samples": [{"x": [16, 16], "label": "wall-ahead"},
{"x": [50, 50], "label": "open"}]}'
model = json.loads(MODEL)
data = [(s["x"], s["label"]) for s in model["samples"]]
print(model["kind"], len(data), data[1][1])
```

Answer:

```
nearest-neighbour 2 open
```

The text becomes a dictionary. The two samples turn back into (features, label) pairs, and the second one's label is open.

4. Why does the saved model include a `kind` field? [1 mark]

- ☐ **A** So the program that loads it knows what to do with it
- ☐ **B** JSON refuses to save without one
- ☐ **C** It records who made the model
- ☐ **D** It makes the text shorter

> **Answer:** A. A nearest-neighbour model and a network need different code. The `kind` says which.

5. How do you get a numpy weights table into a form JSON can write? [1 mark]

- ☐ **A** `W1.tolist()`
- ☐ **B** `json.dumps(W1)` works directly
- ☐ **C** `str(W1)`
- ☐ **D** `W1.shape`

> **Answer:** A. `tolist()` gives plain lists of numbers, and `np.array(...)` turns them back when you load.

6. Which of these can JSON write down as they are? [1 mark]

Tick every answer that is true.

- ☐ **A** A list of numbers
- ☐ **B** A dictionary
- ☐ **C** None
- ☐ **D** A numpy array
- ☐ **E** A tuple, kept as a tuple

> **Answer:** A, B, C. JSON has lists, dictionaries, numbers, strings, True, False and None. Tuples become lists and numpy arrays need `tolist()` first.

The task: a model from text

Load the model in `MODEL`, classify the current view with it, and print `model says: <label>`. Do not drive.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()
import json
MODEL = '{"kind": "nearest-neighbour", "features": "depth grid rows 2 and 3", "samples":
[{"x": [54, 52, 51, 51, 51, 51, 52, 54, 54, 52, 51, 51, 51, 51, 52, 54], "label": "open"},
{"x": [16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16], "label": "wall-
ahead"}, {"x": [36, 47, 46, 46, 46, 16, 16, 16, 36, 47, 46, 46, 46, 16, 16, 16], "label":
"gap-left"}, {"x": [16, 16, 16, 46, 46, 46, 47, 36, 16, 16, 16, 46, 46, 46, 47, 36],
"label": "gap-right"}]}'
model = json.loads(MODEL)
print(model['kind'], len(model['samples']), 'samples')
```

The hint students can ask for: MODEL is a model saved as text. Load it with json.loads, classify the current view with it, and print `model says: <label>`. Do not drive.

A solution

```
from bugbot import *
connect()
def level_rows():
    return tof_grid()[16:32]              # rows 2 and 3: the 16 readings that look
straight ahead

def nearest(sample, data):
    # the label of the recorded sample most like this one (smallest sum of squared
differences)
    best_label, best_d = None, 1e18
    for feats, label in data:
        d = sum((a - b) ** 2 for a, b in zip(feats, sample))
        if d < best_d:
            best_label, best_d = label, d
    return best_label
import json
MODEL = '{"kind": "nearest-neighbour", "features": "depth grid rows 2 and 3", "samples":
[{"x": [54, 52, 51, 51, 51, 51, 52, 54, 54, 52, 51, 51, 51, 51, 52, 54], "label": "open"},
{"x": [16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16], "label": "wall-
ahead"}, {"x": [36, 47, 46, 46, 46, 16, 16, 16, 36, 47, 46, 46, 46, 16, 16, 16], "label":
"gap-left"}, {"x": [16, 16, 16, 46, 46, 46, 47, 36, 16, 16, 16, 46, 46, 46, 47, 36],
"label": "gap-right"}]}'
model = json.loads(MODEL)
data = [(s["x"], s["label"]) for s in model["samples"]]
print("model says:", nearest(level_rows(), data))
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.