Introduction to Tensorflow

Dr. Xiaowei Huang

https://cgi.csc.liv.ac.uk/~xiaowei/

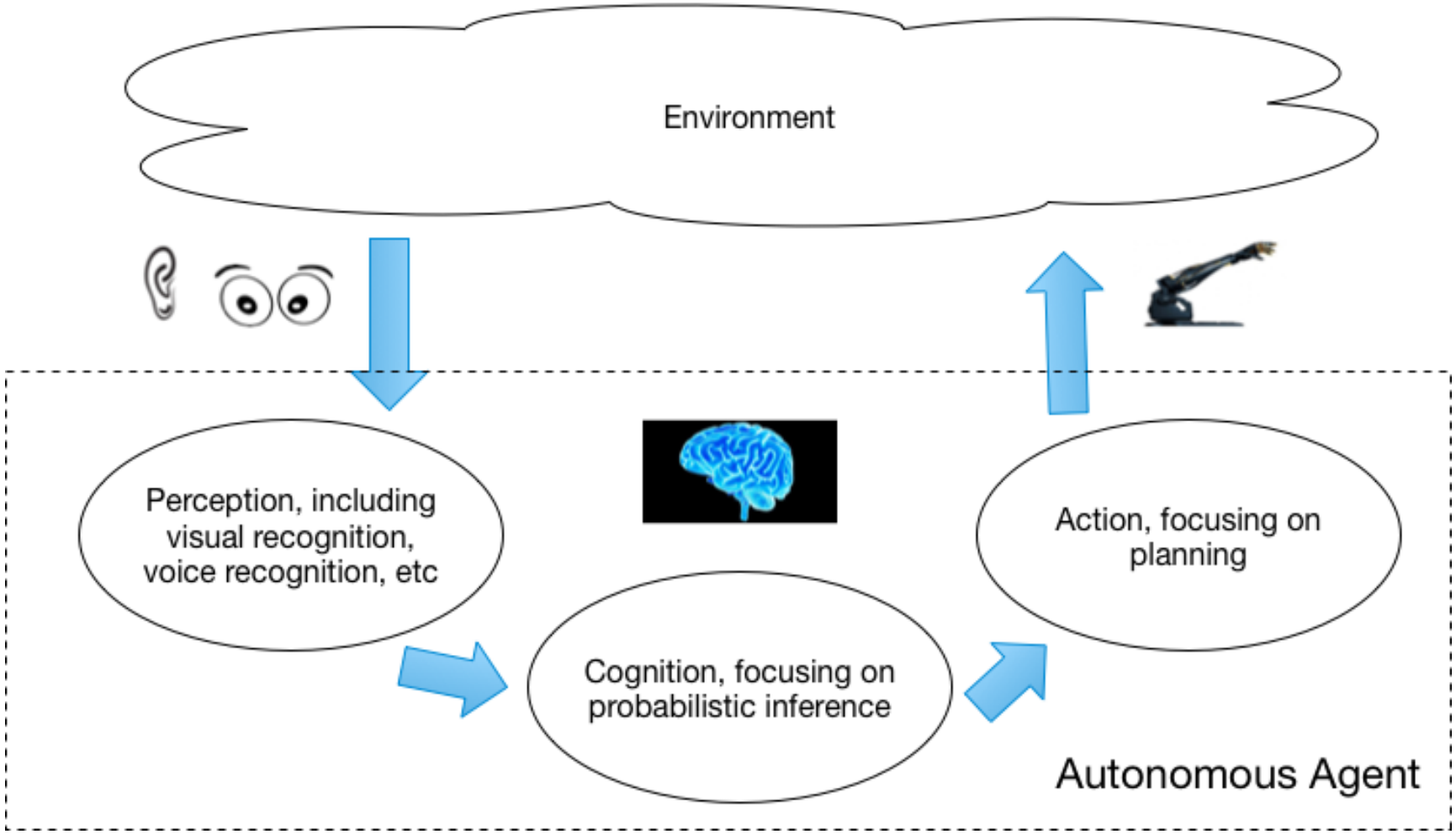
Up to now,

- Overview of Machine Learning
- Traditional Machine Learning Algorithms
  - Decision tree learning (hypothesis: tree, preference: smaller trees, learning algorithm: by entropy aiming to reduce uncertainty)
  - K-nn (definition, performance problem: k-d trees)
  - Model evaluation (confusion matrix, ROC and PR curves)
  - Linear/logistic regression (hypothesis: linear model, preference: objectives/loss functions, why logistic?)
  - Gradient descent (idea, how to compute gradient, the iterative algorithm)
  - Naïve Bayes (assumption, how is it used to reduce the number of parameters)

What’s left?

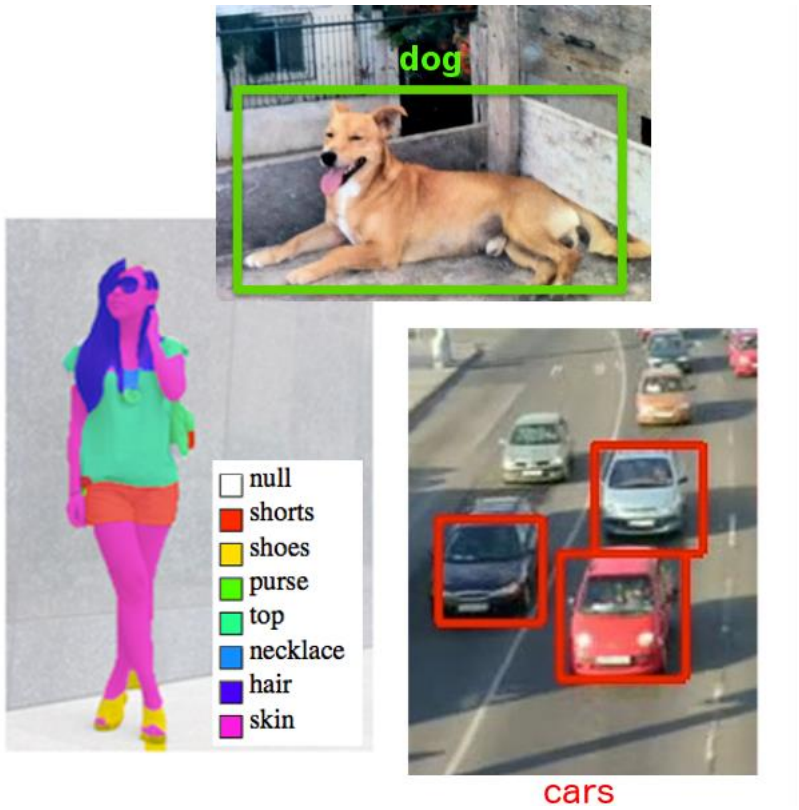
- Deep learning
- Probabilistic graphical model

Perception-Cognition-Action Loop



Teaching content:
traditional learning,
deep learning

Perception

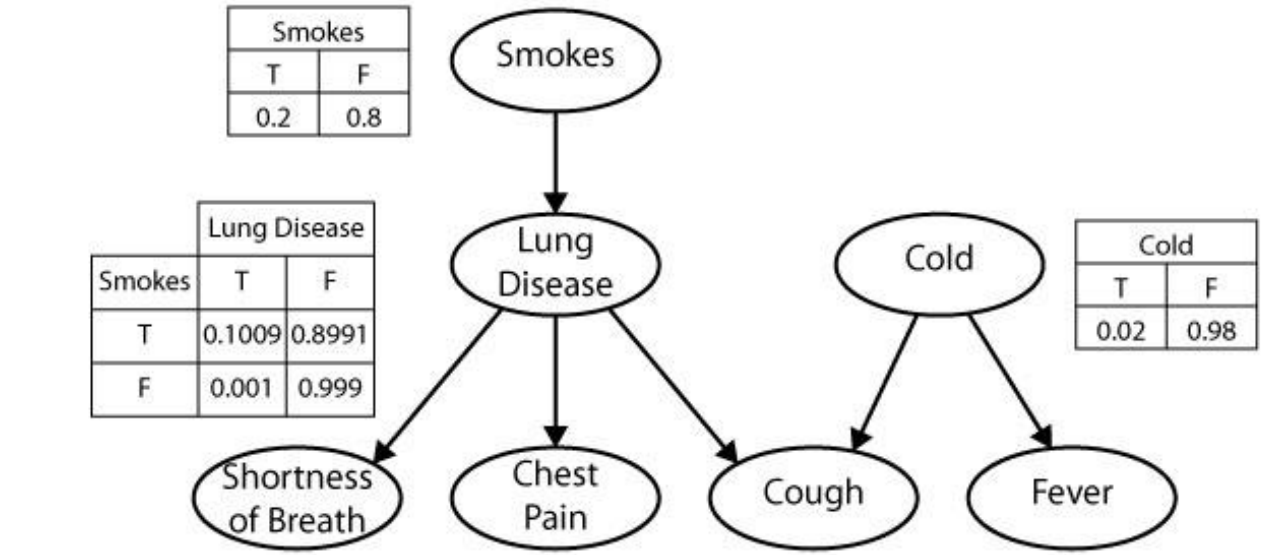


Visual Recognition

Voice Recognition

…

Cognition by Probabilistic Inference



Smokes	
T	F
0.2	0.8

	Lung Disease	
Smokes	T	F
T	0.1009	0.8991
F	0.001	0.999

Cold	
T	F
0.02	0.98

	Shortness of Breath	
Lung Disease	T	F
T	0.208	0.792
F	0.01	0.99

	Chest Pain	
Lung Disease	T	F
T	0.208	0.792
F	0.01	0.99

		Cough	
Lung Disease	Cold	T	F
T	T	0.7525	0.2475
T	F	0.505	0.495
F	T	0.505	0.495
F	F	0.01	0.99

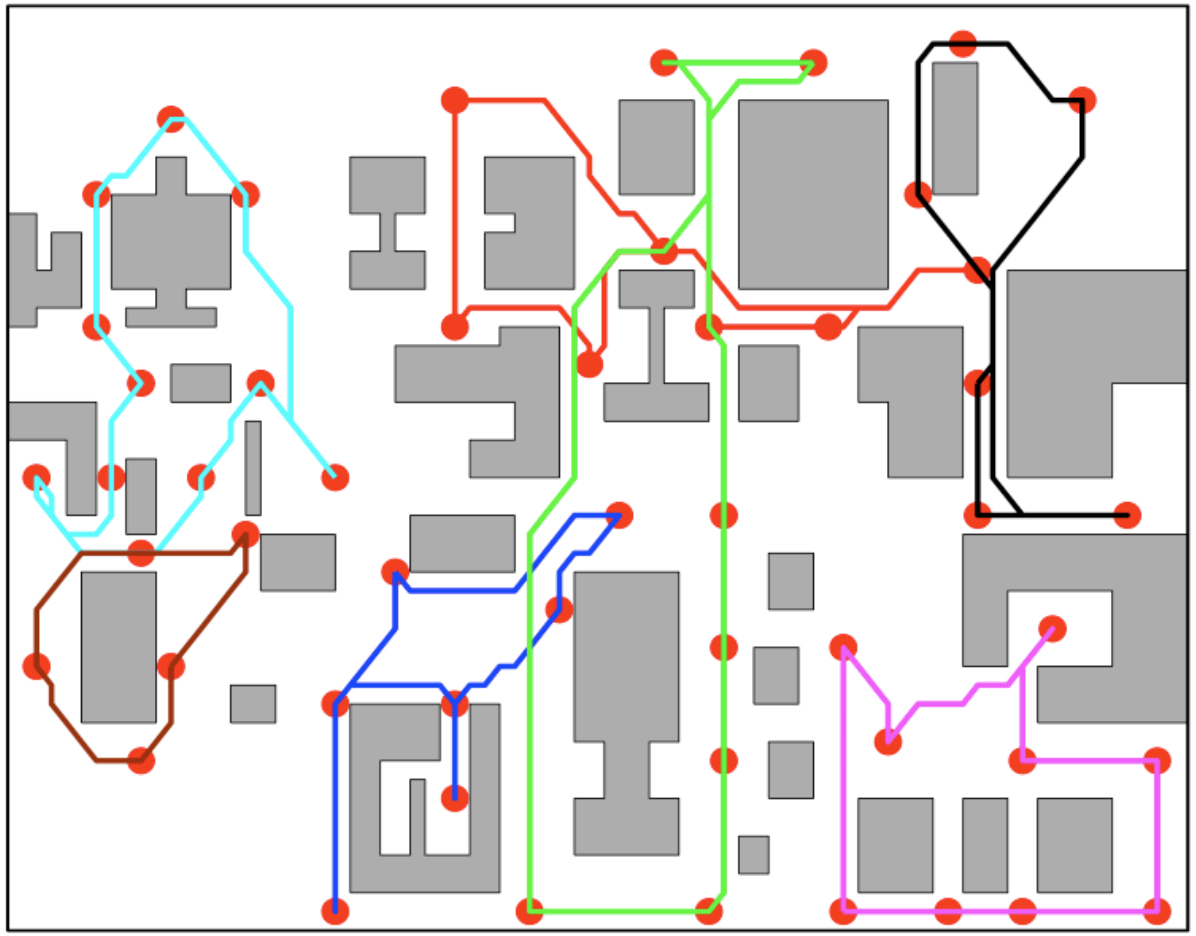
	Fever	
Cold	T	F
T	0.307	0.693
F	0.01	0.99

Q. how to automatically infer the disease (e.g., lung disease, cold, etc) from the symptoms (e.g., smokes, shortness of breath, chest pain, cough, fever, etc)?

Note: Symptoms obtained from perception.

Teaching content:
in other modules, e.g., COMP111, COMP222

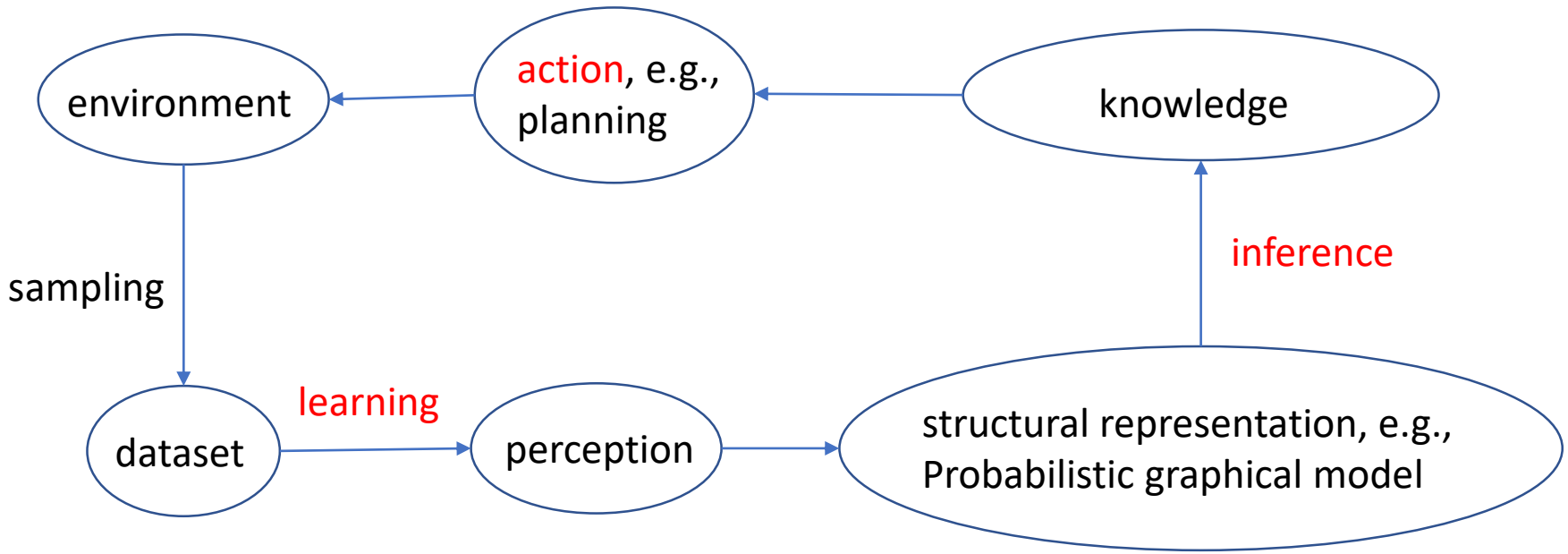
Action by Planning



After cognition, we may use the obtained knowledge to react to the environment

Q: in the factory floor as shown in the left diagram, how many robots is needed to patrol the area? and **how to plan** their activities?

What's left?


environment
action, e.g.,
planning
knowledge
sampling
inference
learning
dataset
perception
structural representation, e.g.,
Probabilistic graphical model

Introduction to Tensorflow

Deep-Learning Package Design Choices

- Model specification:
  - Configuration file (e.g. Caffe, DistBelief, CNTK) versus
  - programmatic generation (e.g. Torch, Theano, Tensorflow)
- For programmatic models, choice of high-level language:
  - Lua (Torch) vs. Python (Theano, Tensorflow) vs others.
- We chose to work with python because of rich community and library infrastructure.

I am going to use this one

I used these two

What is TensorFlow?

- TensorFlow is a deep learning library open-sourced by Google.
- But what does it actually do?
  - TensorFlow provides primitives for defining functions on tensors and automatically computing their derivatives.

But what's a Tensor?

- Formally, tensors are multilinear maps from vector spaces to the real numbers (V vector space, and V* dual space)

$$f : \underbrace{V^* \times \cdots V^*}_{p \text{ copies}} \times \underbrace{V \times \cdots V}_{q \text{ copies}} \to \mathbb{R}$$

A scalar is a tensor ($f : \mathbb{R} \to \mathbb{R},\ f(e_1) = c$)

A vector is a tensor ($f : \mathbb{R}^n \to \mathbb{R},\ f(e_i) = v_i$)

A matrix is a tensor ($f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R},\ f(e_i, e_j) = A_{ij}$)

- Common to have fixed basis, so a tensor can be represented as a multidimensional array of numbers.

TensorFlow vs. Numpy

- Few people make this comparison, but TensorFlow and Numpy are quite similar. (Both are N-d array libraries!)
- Numpy has Ndarray support, but doesn't offer methods to create tensor functions and automatically compute derivatives (+ no GPU support).

VS

Simple Numpy Recap

```
In [23]: import numpy as np

In [24]: a = np.zeros((2,2)); b = np.ones((2,2))

In [25]: np.sum(b, axis=1)
Out[25]: array([ 2.,  2.])

In [26]: a.shape
Out[26]: (2, 2)

In [27]: np.reshape(a, (1,4))
Out[27]: array([[ 0.,  0.,  0.,  0.]])
```

Repeat in TensorFlow

```
In [31]: import tensorflow as tf

In [32]: tf.InteractiveSession()

In [33]: a = tf.zeros((2,2)); b = tf.ones((2,2))

In [34]: tf.reduce_sum(b, reduction_indices=1).eval()
Out[34]: array([ 2.,  2.], dtype=float32)

In [35]: a.get_shape()
Out[35]: TensorShape([Dimension(2), Dimension(2)])

In [36]: tf.reshape(a, (1, 4)).eval()
Out[36]: array([[ 0.,  0.,  0.,  0.]], dtype=float32)
```

More on `Session` soon

More on `.eval()` in a few slides

`TensorShape` behaves like a python tuple.

Numpy to TensorFlow Dictionary

Numpy	TensorFlow
`a = np.zeros((2,2)); b = np.ones((2,2))`	`a = tf.zeros((2,2)), b = tf.ones((2,2))`
`np.sum(b, axis=1)`	`tf.reduce_sum(a,reduction_indices=[1])`
`a.shape`	`a.get_shape()`
`np.reshape(a, (1,4))`	`tf.reshape(a, (1,4))`
`b * 5 + 1`	`b * 5 + 1`
`np.dot(a,b)`	`tf.matmul(a, b)`
`a[0,0], a[:,0], a[0,:]`	`a[0,0], a[:,0], a[0,:]`