
Introduction to Machine Learning with Keras
github.com/mathemakitten/keras-workshop

Hello!

About Me

- data scientist
- mathematics grad
- coffee enthusiast
- women in STEM
- Toronto Women's Data Group organizer
- smart cities: Sidewalk Labs Fellow

www.linkedin.com/in/helen-ngo | helen.ngo@bell.ca | Twitter @mathemakitten

Today's workshop

- What is machine learning?
- What *is* Keras?
- Tutorial 1: Teach a Neural Network to Add
  - We're going to teach a recurrent network how to add without ever explicitly defining the addition function!
- Tutorial 2: Classifying Very Small Images
  - You can learn a lot from 32 x 32 pixels

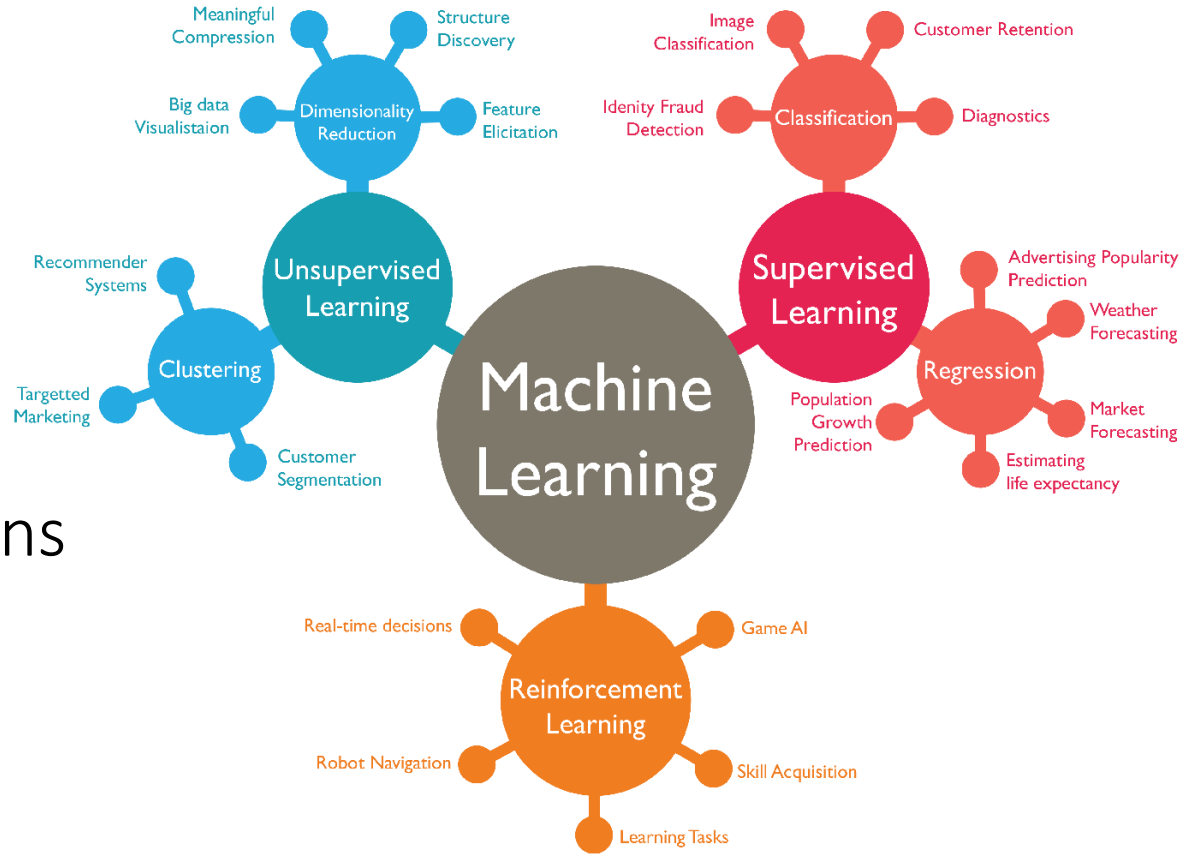
Follow along!

github.com/mathemakitten/keras-workshop

What is machine learning?

Dr. Yoshua Bengio, Université de Montréal:

- Machine learning research is part of research on artificial intelligence
- Seeking to provide knowledge to computers through data, observations and interacting with the world
- That acquired knowledge allows computers to correctly generalize to new settings



What is Keras?

- High-level neural networks API written in Python
- Capable of running on top of TensorFlow, CNTK, or Theano
- Written by François Chollet @ Google
- First release 2015
- Industry users: Netflix, Uber, Yelp, Instacart, Square, NASA

But why should I choose Keras?

- Abstracts away complexity of Tensorflow
- Resilient to change: backends will change over time
- Easy to get started!

What is Google Colab?

- Run Python notebooks in the cloud on Google hardware
- Get access to first-class computation power
- GPU compute—totally free!

- For this workshop, you can use these links on the Git README.md

- Or git clone from the repo and upload it yourself to Colab

```
`git clone git@github.com:mathemakitten/keras-workshop.git`
```

The Unreasonable Effectiveness of Recurrent Neural Networks

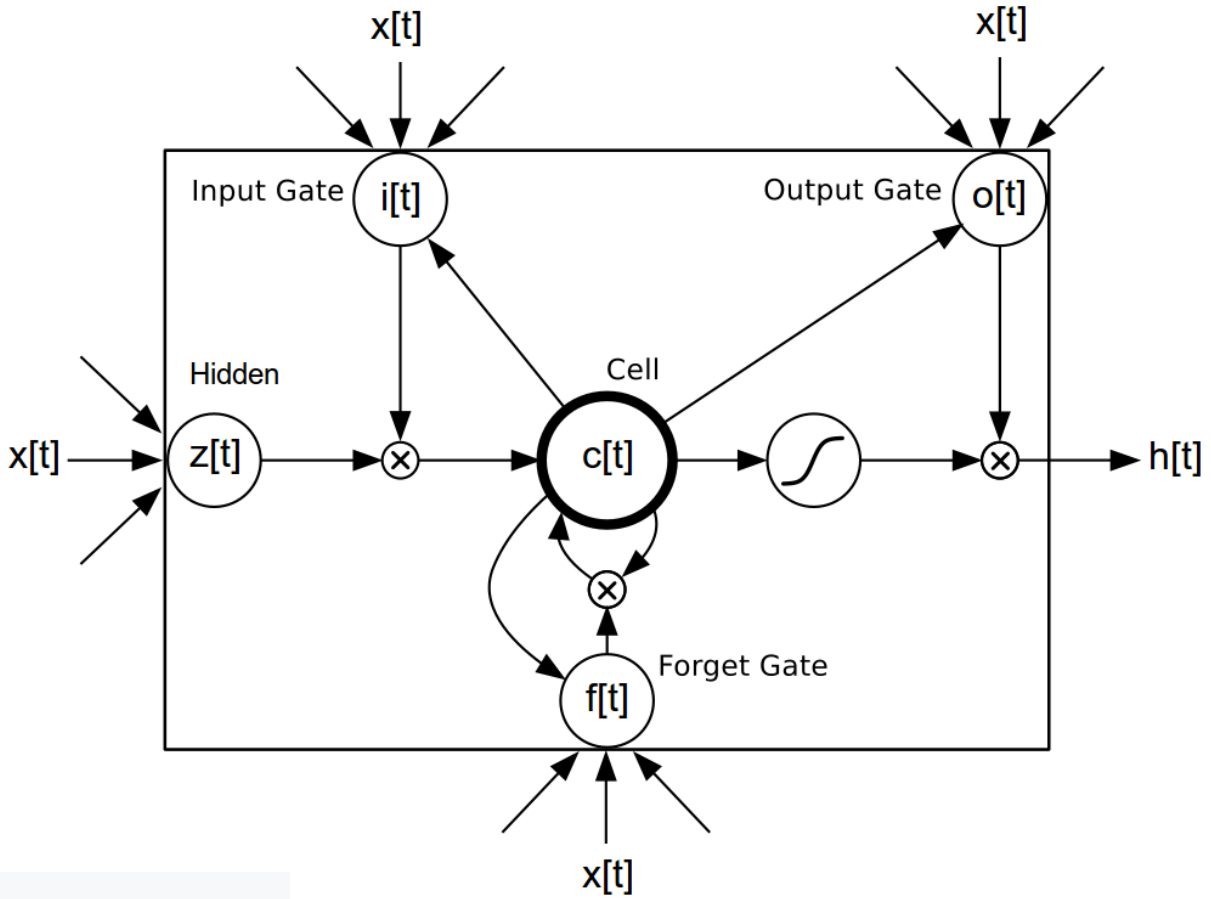
May 21, 2015

There's something magical about Recurrent Neural Networks (RNNs). I still remember when I trained my first recurrent network for Image Captioning. Within a few dozen minutes of training my first baby model (with rather arbitrarily-chosen hyperparameters) started to generate very nice looking descriptions of images that were on the edge of making sense. Sometimes the ratio of how simple your model is to the quality of the results you get out of it blows past your expectations, and this was one of those times. What made this result so shocking at the time was that the common wisdom was that RNNs were supposed to be difficult to train (with more experience I've in fact reached the opposite conclusion). Fast forward about a year: I'm training RNNs all the time and I've witnessed their power and robustness many times, and yet their magical outputs still find ways of amusing me. This post is about sharing some of that magic with you.

> *We'll train RNNs to generate text character by character and ponder the question "how is that even possible?"*

What's an LSTM?

¯_(ツ)_/¯



$$i[t] = \sigma(W[x \to i]x[t] + W[h \to i]h[t-1] + W[c \to i]c[t-1] + b[1 \to i]) \quad (1)$$
$$f[t] = \sigma(W[x \to f]x[t] + W[h \to f]h[t-1] + W[c \to f]c[t-1] + b[1 \to f]) \quad (2)$$
$$z[t] = \tanh(W[x \to c]x[t] + W[h \to c]h[t-1] + b[1 \to c]) \quad (3)$$
$$c[t] = f[t]c[t-1] + i[t]z[t] \quad (4)$$
$$o[t] = \sigma(W[x \to o]x[t] + W[h \to o]h[t-1] + W[c \to o]c[t] + b[1 \to o]) \quad (5)$$
$$h[t] = o[t]\tanh(c[t]) \quad (6)$$

Source: https://github.com/Element-Research/rnn