Neural Networks for Machine Learning demonstrations
Neural Network Architectures

Current focus on large networks with different “architectures” suited for different kinds of tasks

• Feedforward Neural Network
• CNN: Convolutional Neural Network
• RNN: Recurrent Neural Network
• LSTM: Long Short Term Memory
• GAN: Generative Adversarial Network
Feedforward Neural Network

• Connections allowed from a node in layer $i$ only to nodes in layer $i+1$
  i.e., no cycles or loops

• Simple, widely used architecture.

HTTP://PLAYGROUND.TENSORFLOW.ORG/
Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.

DATA
Which dataset do you want to use?

FEATURES
Which properties do you want to feed in?

OUTPUT
Test loss 0.435
Training loss 0.432

Colors show data, neuron and weight values.

Show test data  Discretize output

---

HTTP://PLAYGROUND.TENSORFLOW.ORG/
CNN: Convolutional Neural Network

- Good for image processing: classification, object recognition, automobile lane tracking, etc.
- Classic demo: learn to recognize hand-written digits from MNIST data with 70K examples
RNN: Recurrent Neural Networks

- Good for learning over sequences of data, e.g., a sentence or words
- LSTM (Long Short Term Memory) a popular architecture

Output so far:
Machine

gif from Adam Geitgey
Deep Learning Frameworks

• Popular open source deep learning frameworks use Python at top-level; C++ in backend
  – TensorFlow (via Google)
  – PyTorch (via Facebook)
  – MxNet (Apache)
  – Caffe (Berkeley)
• Keras: popular API works with the first two and provides good support at architecture level
Scikit-learn

• We’ll look at using scikit-learn’s feed forward model on the iris dataset
Classifying digits with convolutional neural networks

This notebook contains the solution to the MNIST activity.

Load the data

Both Keras and TF-Learn contain the MNIST dataset that can be quickly loaded with some helper functions. This solution will use TF-Learn but the Keras solution will be commented out. The two libraries are very similar.

```
In [1]:

import numpy as np
import keras
from keras.datasets import mnist

# Load data from Keras
(X_train, y_train), (X_test, y_test) = mnist.load_data()

# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```
Sentiment analysis with Recurrent Neural Networks

For this particular dataset a shallow method like tf-idf features into logistic regression will outperform the RNN. But, what this will illustrate is just how simple it is to implement an RNN for sentiment analysis with Keras and TF-Learn. The notebook was run with Keras and the equivalent TF-Learn code will be commented out.

Load the packages

```python
import numpy as np
from keras.preprocessing import sequence
from keras.utils import np_utils
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Embedding, GRU
from keras.datasets import imdb
```

```python
#import tflearn
#from tflearn.data_utils import to_categorical, pad_sequences
#from tflearn.datasets import imdb
```
09 Neural Networks