“Hello world” of deep learning
Keras

If you want to learn Theano:

Very flexible
Need some effort to learn

Easy to learn and use
(still have some flexibility)
You can modify it if you can write TensorFlow or Theano

Interface of TensorFlow or Theano
Keras

• François Chollet is the author of Keras.
  • He currently works for Google as a deep learning engineer and researcher.

• Keras means horn in Greek

• Documentation: [http://keras.io/](http://keras.io/)

• Example: [https://github.com/fchollet/keras/tree/master/examples](https://github.com/fchollet/keras/tree/master/examples)
使用 Keras 心得
“Hello world”

- Handwriting Digit Recognition

MNIST Data: http://yann.lecun.com/exdb/mnist/
Keras provides data sets loading function: http://keras.io/datasets/
Keras

Step 1: define a set of function

Step 2: goodness of function

Step 3: pick the best function

model = Sequential()

model.add(Dense(input_dim=28*28, output_dim=500))
model.add(Activation('sigmoid'))
softplus, softsign, relu, tanh, hard_sigmoid, linear

model.add(Dense(output_dim=500))
model.add(Activation('sigmoid'))

model.add(Dense(output_dim=10))
model.add(Activation('softmax'))
Keras

Several alternatives: https://keras.io/objectives/

```python
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```
Keras

**Step 1:** Define a set of function

**Step 2:** Goodness of function

**Step 3:** Pick the best function

---

**Step 3.1: Configuration**

```python
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, Nadam

---

**Step 3.2: Find the optimal network parameters**

```python
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```

---

Training data (Images)  |  Labels (digits)  |  In the following slides
Keras

Step 1: define a set of function

Step 2: goodness of function

Step 3: pick the best function

Step 3.2: Find the optimal network parameters

```
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```

Number of training examples

28 x 28 = 784

Number of training examples

10
Mini-batch

Randomly initialize network parameters

- Pick the 1\textsuperscript{st} batch
  \[ L' = l_1^1 + l_3^{16} + \ldots \]
  Update parameters once

- Pick the 2\textsuperscript{nd} batch
  \[ L'' = l_2^2 + l_{16}^{16} + \ldots \]
  Update parameters once

- Until all mini-batches have been picked

We do not really minimize total loss!

one epoch

Repeat the above process
Mini-batch

Batch size influences both *speed* and *performance*. You have to tune it.

```python
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```

100 examples in a mini-batch

Batch size = 1 ➔ Stochastic gradient descent

Repeat 20 times ➔ one epoch

➢ Pick the 1st batch

\[ L' = l^1 + l^{31} + \ldots \]

Update parameters once

➢ Pick the 2nd batch

\[ L'' = l^2 + l^{16} + \ldots \]

Update parameters once

➢ Until all mini-batches have been picked
Speed

- Smaller batch size means more updates in one epoch
  - E.g. 50000 examples
  - batch size = 1, 50000 updates in one epoch
  - batch size = 10, 5000 updates in one epoch

Very large batch size can yield worse performance

Batch size = 1 and 10, update the same amount of times in the same period.
Batch size = 10 is more stable, converge faster

GTX 980 on MNIST with 50000 training examples
Speed - Matrix Operation

\[ y = f(x) \] Forward pass (Backward pass is similar)

\[ y = \sigma(W^L \cdots \sigma(W^2 \sigma(W^1 x + b^1) + b^2) \cdots + b^L) \]
Speed - Matrix Operation

• Why mini-batch is faster than stochastic gradient descent?

Stochastic Gradient Descent

\[ z^1 = W^1 x \]

Mini-batch

\[ \begin{align*}
    z^1 & = z^1 \\
    z^1 & = z^1 \\
\end{align*} \]

Practically, which one is faster?
Keras

Save and load models

http://keras.io/getting-started/faq/#how-can-i-save-a-keras-model

How to use the neural network (testing):

```python
score = model.evaluate(x_test, y_test)
print('Total loss on Testing Set:', score[0])
print('Accuracy of Testing Set:', score[1])
```

case 2:
```
result = model.predict(x_test)
```
Keras

• Using GPU to speed training
  • Way 1
    • THEANO_FLAGS=device=gpu0 python YourCode.py
  • Way 2 (in your code)
    • import os
    • os.environ['THEANO_FLAGS'] = "device=gpu0"