“Hello world” of deep learning

Courtesy of Prof. Lee, Hung-Yi for the slides

Keras

If you want to learn theano:

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Theano%20DNN.ecm.mp4/index.html

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RNN%20training%20(v6).ecm.mp4/index.html

or theano

Very flexible

Need some effort to learn

Interface of TensorFlow or Theano

keras

Easy to learn and use

(still have some flexibility)

You can modify it if you can write TensorFlow or Theano

Keras

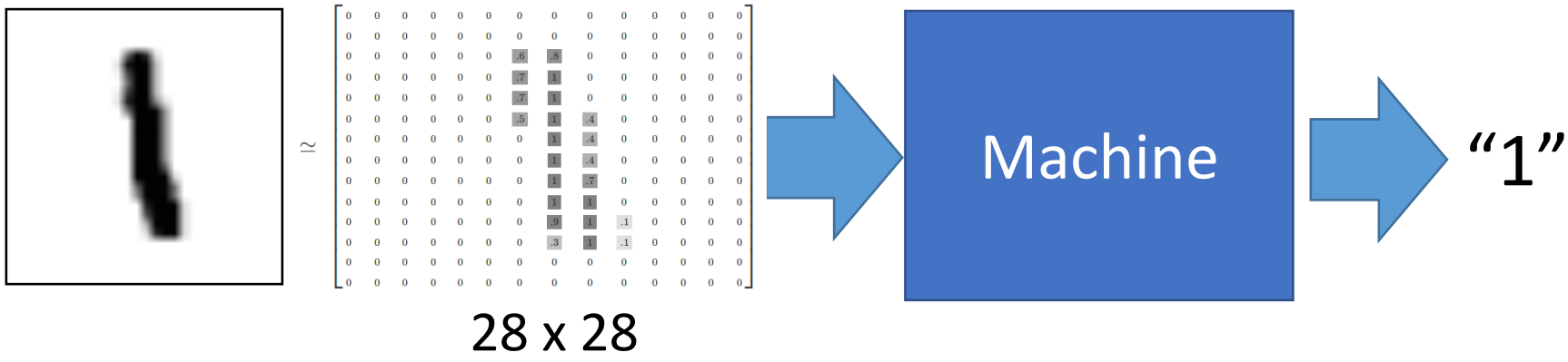
- François Chollet is the author of Keras.
  - He currently works for Google as a deep learning engineer and researcher.
- Keras means *horn* in Greek
- Documentation: http://keras.io/
- Tutorial: (Immerse on Youtube)
  Deep Learning with Python, TensorFlow, and Keras tutorial
  https://www.youtube.com/watch?v=wQ8BIBpya2k
- Example:
  https://github.com/fchollet/keras/tree/master/examples

感謝 沈昇勳 同學提供圖檔

使用 Keras 心得

“Hello world”

- Handwriting Digit Recognition



MNIST Data: http://yann.lecun.com/exdb/mnist/

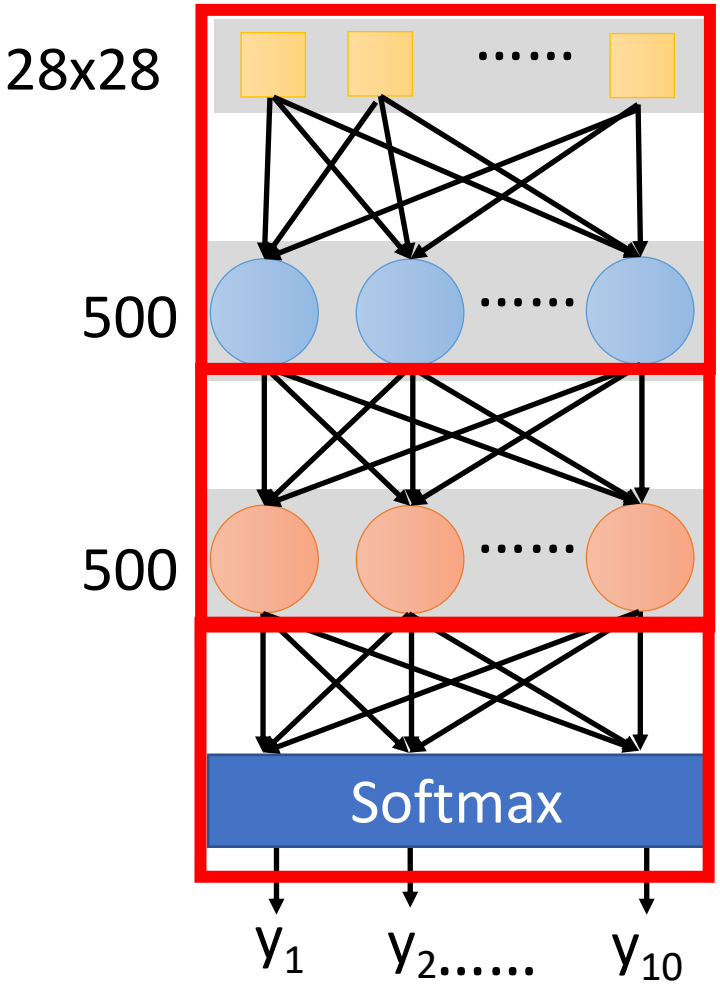
Keras provides data sets loading function: http://keras.io/datasets/

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') )
```

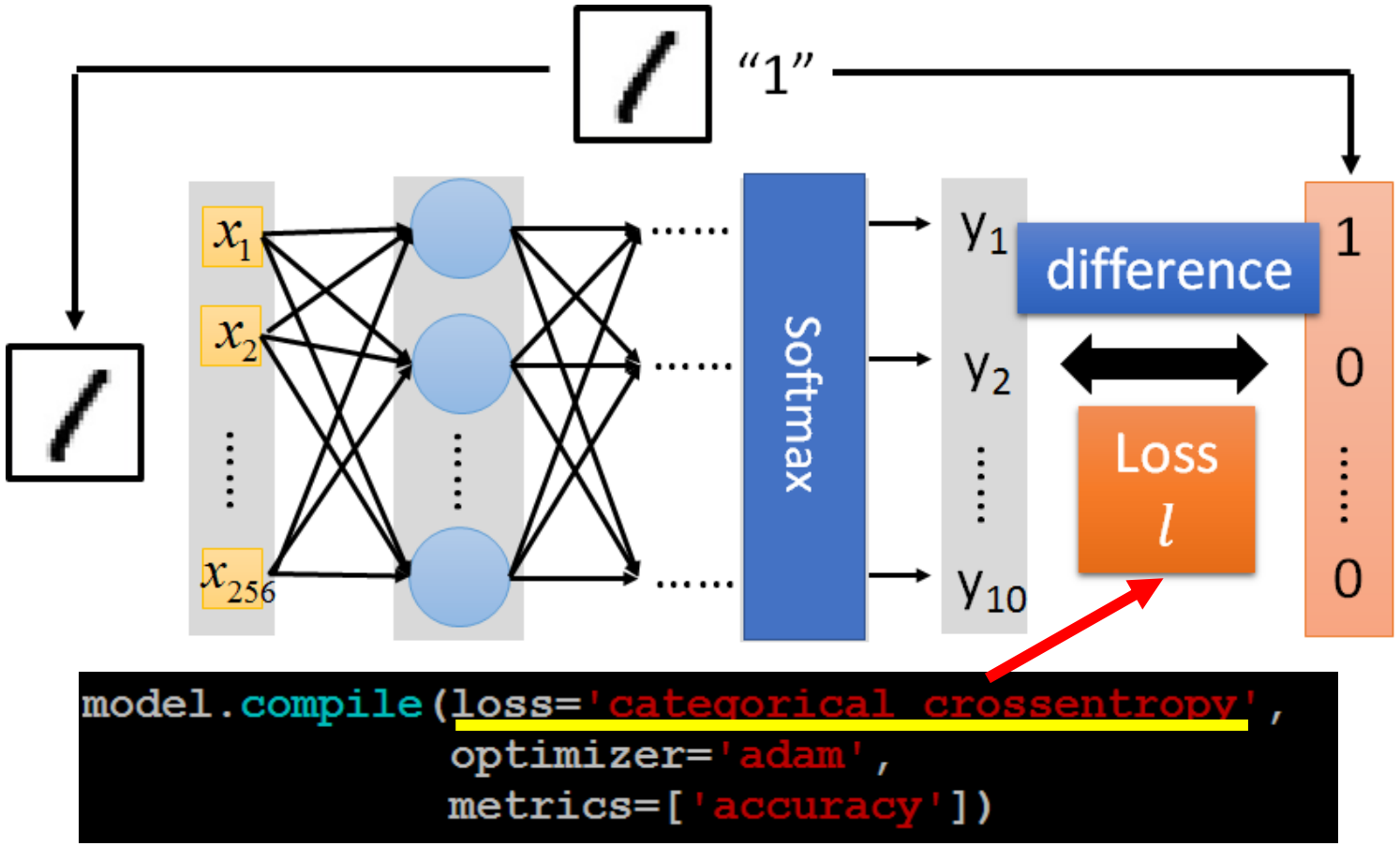
```
model.add( Dense(output_dim=10 ) )
model.add( Activation('softmax') )
```

Keras

Step 1: define a set of function

Step 2: goodness of function

Step 3: pick the best function




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

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

Keras

Step 3.1: Configuration

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

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

Step 3.2: Find the optimal network parameters

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

Training data (Images)

Labels (digits)

In the following slides

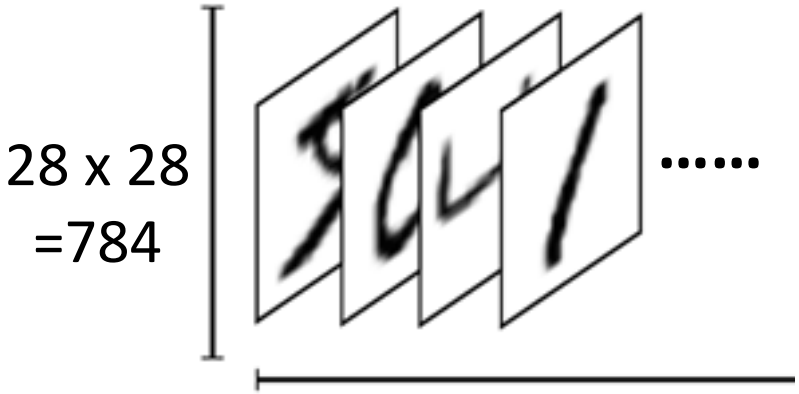
Keras

Step 1: define a set of function → Step 2: goodness of function → Step 3: pick the best function

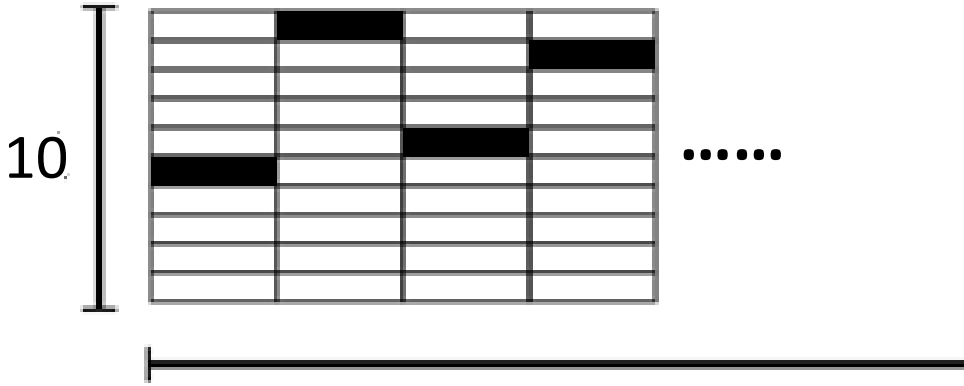
Step 3.2: Find the optimal network parameters

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

numpy array



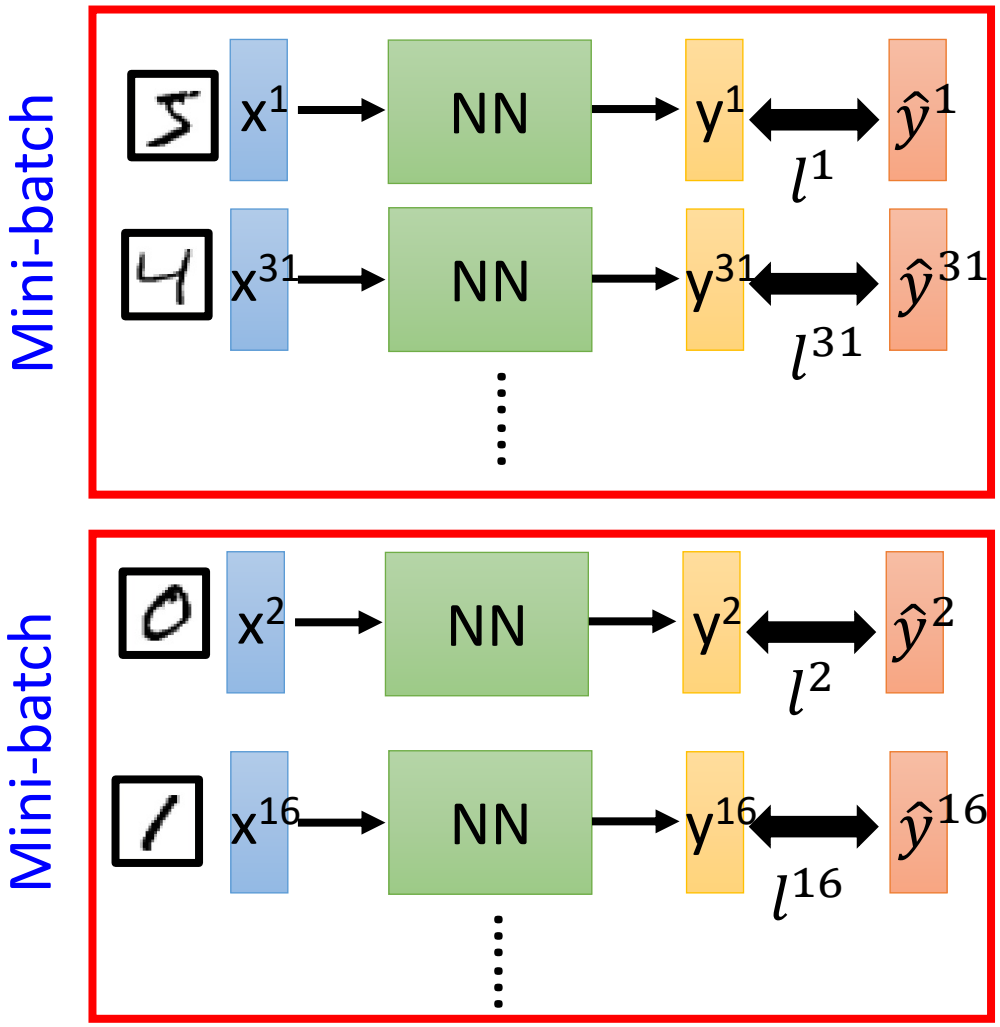
numpy array



https://www.tensorflow.org/versions/r0.8/tutorials/mnist/beginners/index.html

We do not really minimize total loss!

Mini-batch



- Randomly initialize network parameters
- Pick the 1st batch

  $L' = l^1 + l^{31} + \cdots$

  Update parameters once
- Pick the 2nd batch

  $L'' = l^2 + l^{16} + \cdots$

  Update parameters once

  ⋮
- Until all mini-batches have been picked

one epoch

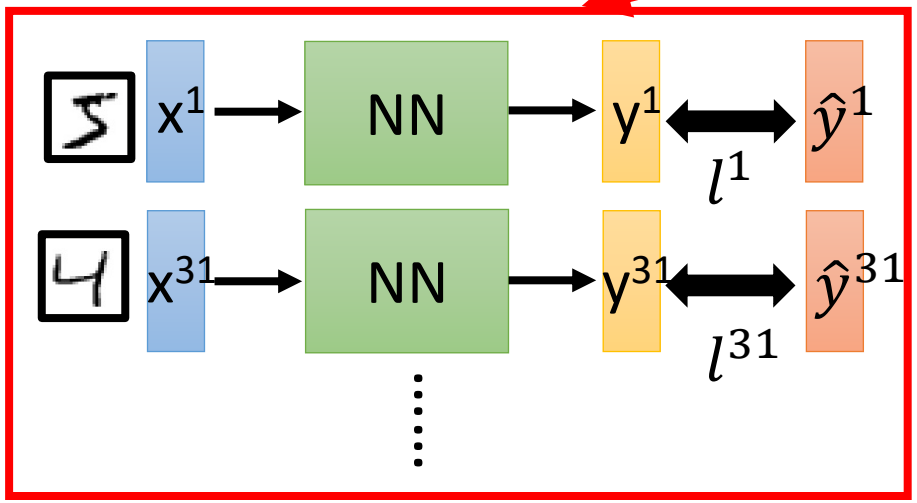
Repeat the above process

Mini-batch

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

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

Mini-batch



100 examples in a mini-batch

Batch size = 1 ➡

Stochastic gradient descent

- Pick the 1st batch

  $L' = l^1 + l^{31} + \cdots$

  Update parameters once

- Pick the 2nd batch

  $L'' = l^2 + l^{16} + \cdots$

  Update parameters once

  ⋮

- Until all mini-batches have been picked

Repeat 20 times

one epoch

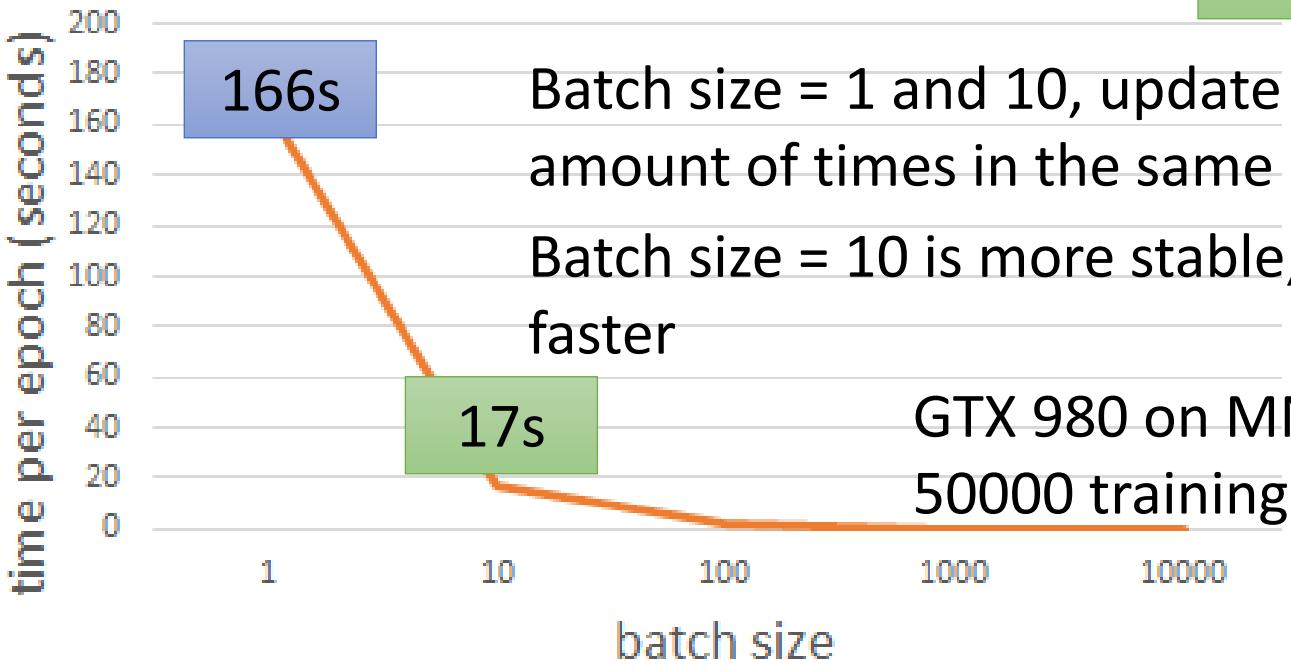
Speed

Very large batch size can yield worse performance

- 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

166s 1 epoch

17s 10 epoch

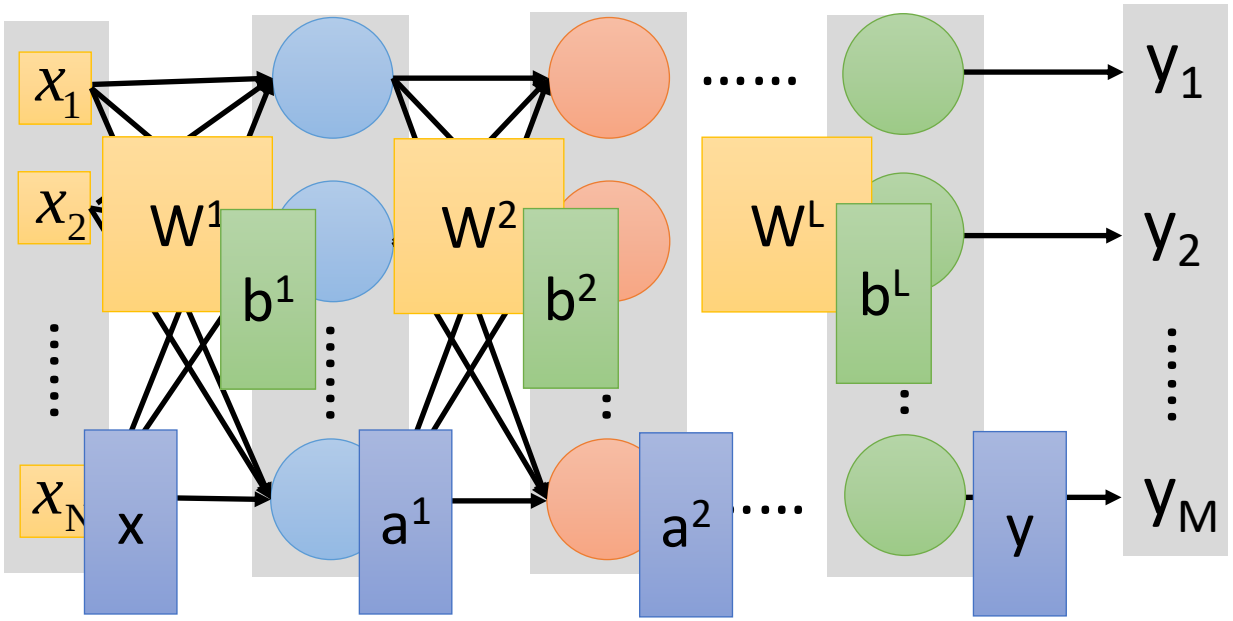


Batch size = 1 and 10, update the same amount of times in the same period.

Batch size = 10 is more stable, converge faster

GTX 980 on MNIST with 50000 training examples

Speed - Matrix Operation



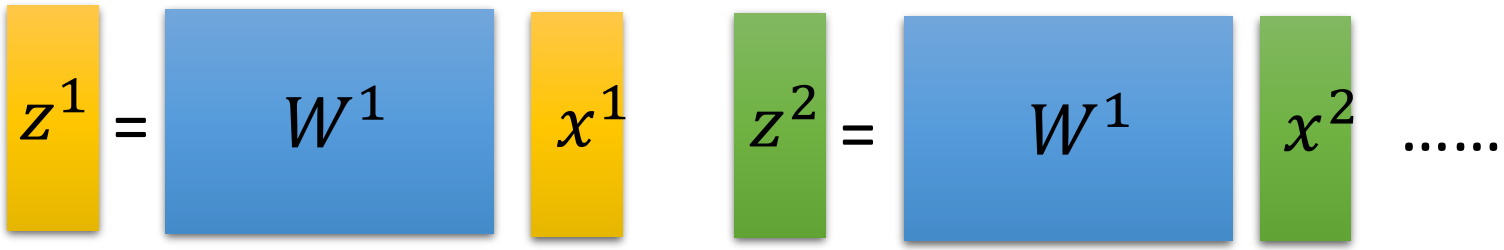
$$\mathbf{y} = f(\mathbf{x})$$ Forward pass (Backward pass is similar)

$$= \sigma(\mathbf{W}^L \cdots \sigma(\mathbf{W}^2 \sigma(\mathbf{W}^1 \mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2) \cdots + \mathbf{b}^L)$$

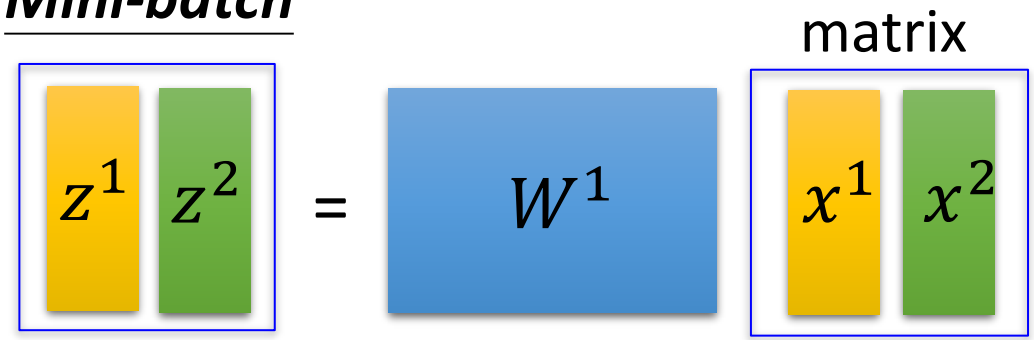
Speed - Matrix Operation

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

Stochastic Gradient Descent

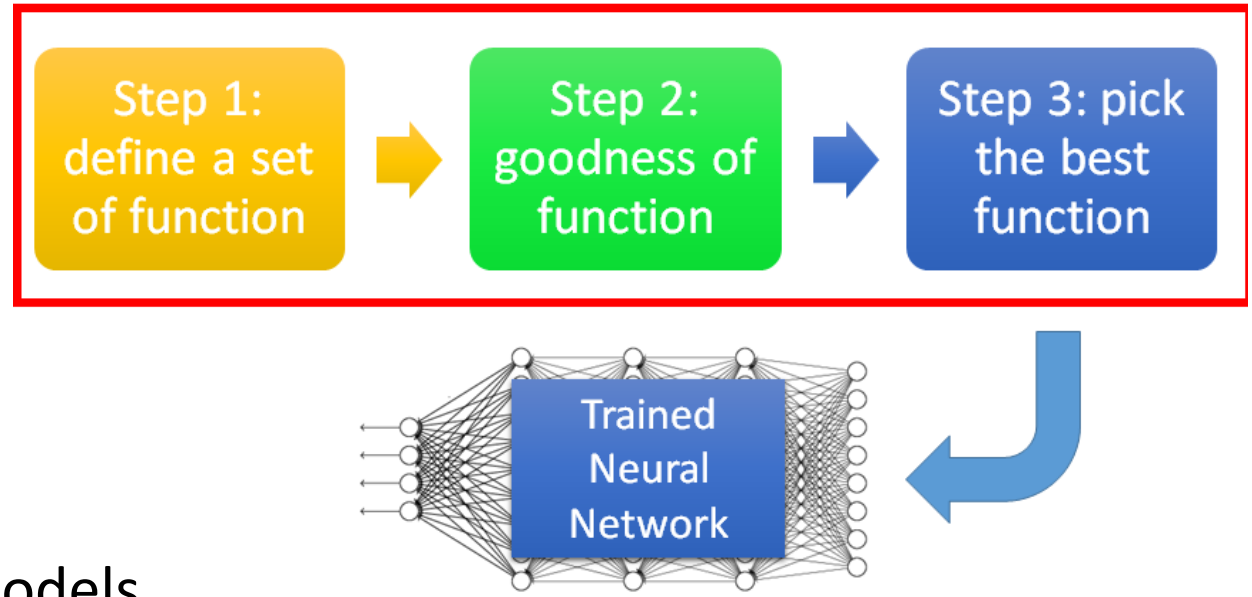


Mini-batch



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):

case 1:

```
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)
```