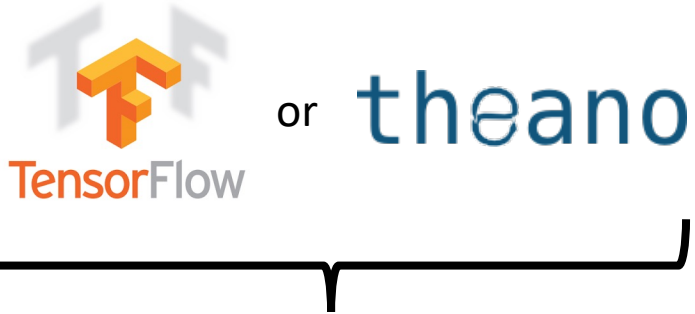
Keras

Keras



Very flexible

Need some effort to learn

Interface of TensorFlow or Theano

keras

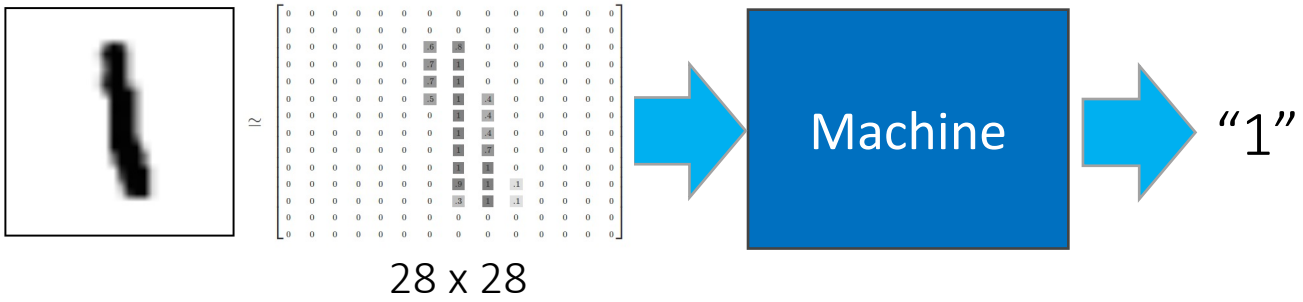
Easy to learn and use

(still have some flexibility)

You can modify it if you can write TensorFlow or Theano

“Hello World”

Handwriting Digit Recognition

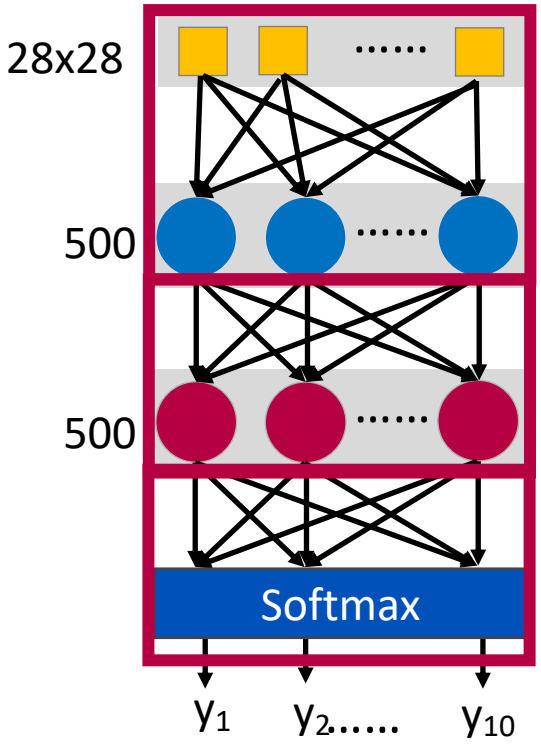


MNIST Data: http://yann.lecun.com/exdb/mnist/

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

Keras: Part I

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') )
```

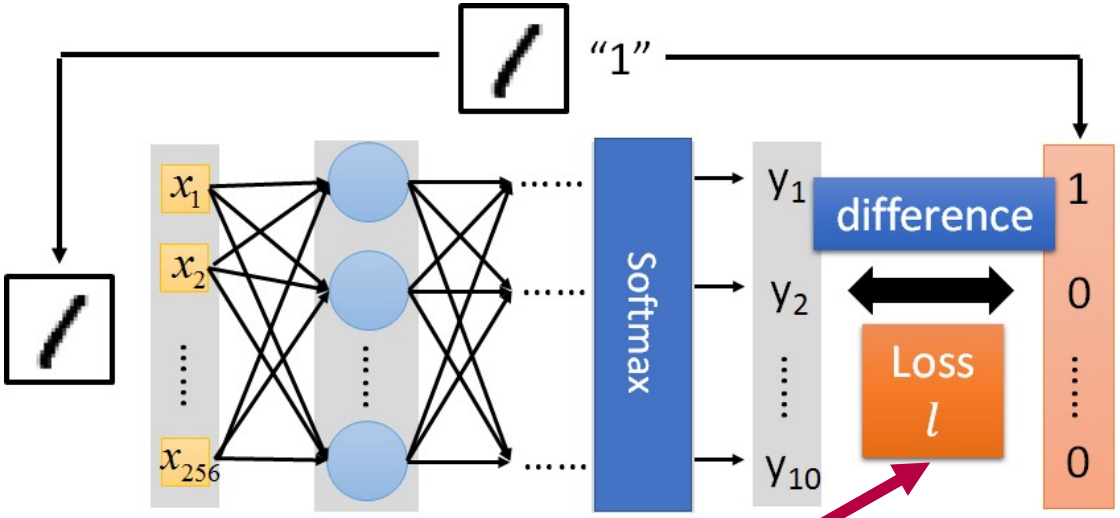
softplus, softsign, relu, tanh, hard_sigmoid, linear

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

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

Keras: Part II

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: Part III

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: Part IV

Step 3.2: Find the optimal network parameters

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

numpy array

28 x 28 =784

.....

Number of training examples

numpy array

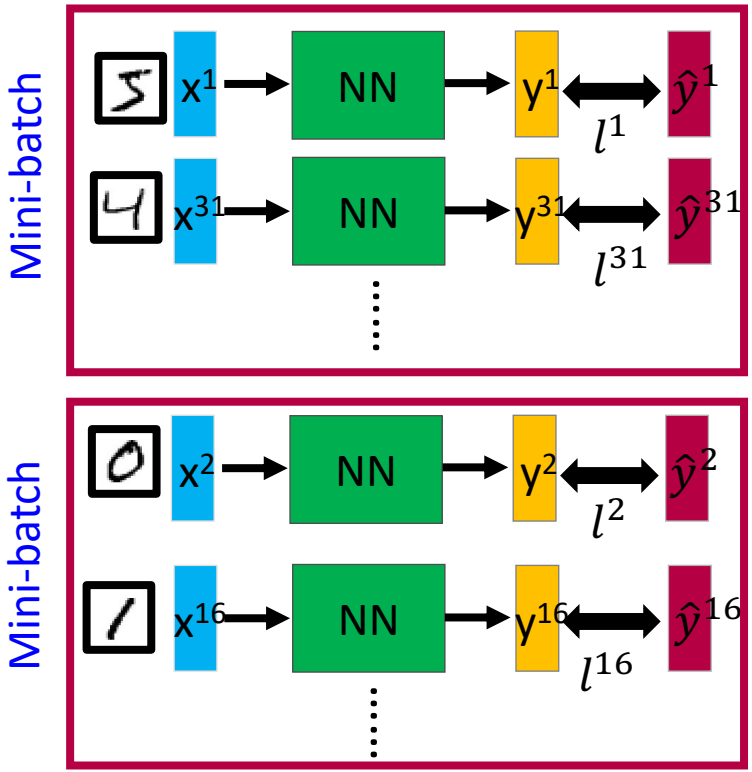
10

.....

Number of training examples

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

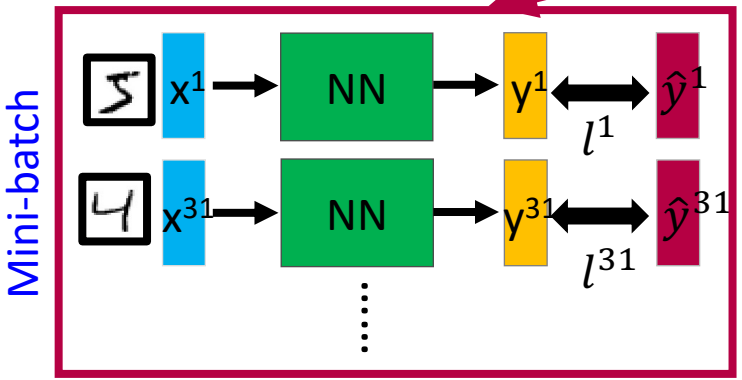
one epoch

Repeat the above process

Mini-Batch, Continued

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

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



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

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

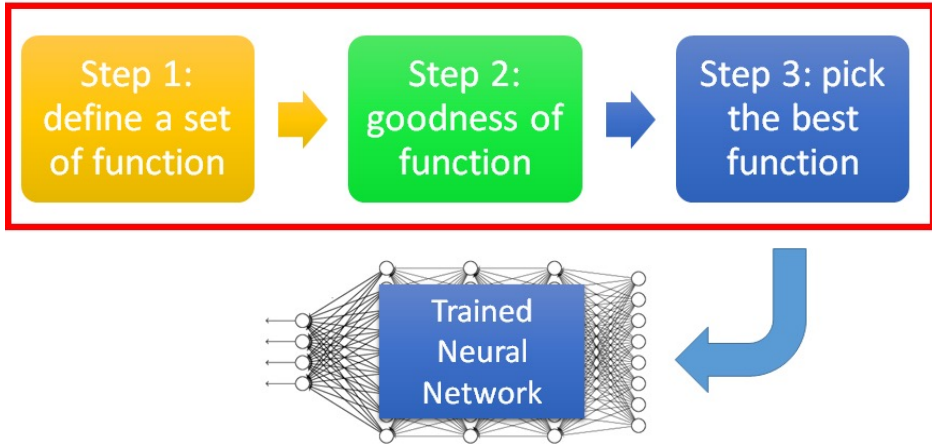
  Update parameters once

  ⋮

- Until all mini-batches have been picked

Repeat 20 times — one epoch

Keras: Part V



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)
```