

Deep learning & applications

Practice#3

Tae Hyun Kim

Tensorflow

- Official tensorflow website
 - <https://www.tensorflow.org/>
- Both CPU/GPU versions are available
 - <https://www.tensorflow.org/install>
- Free GPU?
 - Colab
 - **Colab** is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud.
 - For non-commercial (educational) use only.
 - Allow to develop deep learning applications using popular libraries such as **Keras**, **TensorFlow**, **PyTorch**, and **OpenCV**
 - <https://colab.research.google.com/notebooks/gpu.ipynb>

```
# Current stable release for CPU and GPU
$ pip install tensorflow
```

Tensorflow

- Quick guide
 - <https://www.tensorflow.org/tutorials>
- Provides API for Beginners as well as Experts

For beginners

The best place to start is with the user-friendly Keras sequential API. Build models by plugging together building blocks. After these tutorials, read the [Keras guide](#).

Beginner quickstart

This "Hello, World!" notebook shows the Keras Sequential API and `model.fit`.

Keras basics

This notebook collection demonstrates basic machine learning tasks using Keras.

Load data

These tutorials use `tf.data` to load various data formats and build input pipelines.

For experts

The Keras functional and subclassing APIs provide a define-by-run interface for customization and advanced research. Build your model, then write the forward and backward pass. Create custom layers, activations, and training loops.

Advanced quickstart

This "Hello, World!" notebook uses the Keras subclassing API and a custom training loop.

Customization

This notebook collection shows how to build custom layers and training loops in TensorFlow.

Distributed training

Distribute your model training across multiple GPUs, multiple machines or TPUs.

Tensorflow and Keras (from Wiki)

- **Keras:** open source neural network library written in Python
 - Capable of running on top of Tensorflow, Theano, and so on
 - Enable fast experimentation with DNN
 - Primary author and maintainer is François Chollet (Googler)
 - Author of the Xception deep neural network model.
 - In 2017, Google's TensorFlow team decided to support Keras in TF
 - Chollet explained that Keras was conceived to be an interface rather than a standalone ML framework
 - Offers a higher-level, more intuitive set of abstractions
 - make it easy to develop deep learning models regardless of the computational backend used

Quick start

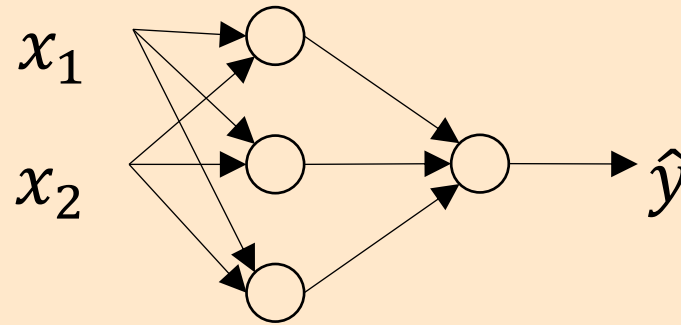
- <https://www.tensorflow.org/tutorials/quickstart/beginner>

Task3: binary classification using wide 2-layered net (cross-entropy loss)

Input: 2-dim vector, $\mathbf{x} = \{x_1, x_2\}$

Output: label of the input, $\mathbf{y} \in \{0,1\}$

Pseudo code #should implement with tensorflow



Step 1. Load pre-generated ' m ' train samples, ' n ' test samples in practice#2

Step 2. Update *params* with ' m ' samples for ' K ' iterations: # K grad updates!

Step 2-1. print W , b every 50 iterations

Step 2-2. calculate the cost on the ' m ' train samples!

Step 2-3. calculate the cost with the ' n ' test samples!

Step 2-4. print accuracy for the ' m ' train samples! (display the number of correctly predicted outputs/ $m \cdot 100$)

Step 2-5. print accuracy with the ' n ' test samples! (display the number of correctly predicted outputs/ $n \cdot 100$)

Report

- Submission due: (5/28, 1pm)
 - Late submission will not be counted
- Submissions: (through LMS system)
 - 1 source file: task3.py
 - Single page pdf: studentid_name.pdf
 - Should not be more than 3 pages
 - Should include
 - tables in the next pages
 - Discussion (what you've learned from the experiments)

Table 1

- Compare loss functions
 - how to set?
 - <https://keras.io/api/losses/>
 - Use 'SGD' optimizer

	BinaryCrossentropy	MeanSquaredError
Accuracy (with train set)		
Accuracy (with test set)		
Train time [sec]		

Table 2

- Compare optimizers
 - how to set?
 - <https://keras.io/api/optimizers/>
- Use the binary cross-entropy loss

	SGD result	RMSProp	Adam
Accuracy (with train set)			
Accuracy (with test set)			
Train time [sec]			

Table 3

- Compare mini-batches
 - Use 'SGD' optimizer
 - Use the binary cross-entropy loss

	Mini-batch = 1	Mini-batch = 32	Mini-batch = 128
Accuracy (with train set)			
Accuracy (with test set)			
Train time [sec]			
Inference (test) time [sec]			