# ENGR-E 533 “Deep Learning Systems” Lecture 03: Deep Learning Toolboxes


**Minje Kim**

Department of Intelligent Systems Engineering

Email: minje@indiana.edu

Website: http://minjekim.com

Research Group: http://saige.sice.indiana.edu

Meeting Request: http://doodle.com/minje



INDIANA UNIVERSITY

SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

# Intro to the GPU Resources

- A little bit about GPU computing
  - GPUs are different from CPUs in terms of their number of cores
    - Tesla V100 has 5120 FP32 cores
    - A better choice for a task with many simple sub-tasks, such as matrix multiplication
    - Each core is much less powerful than a usual CPU core
- Common packages you might need
  - Tensorflow or PyTorch
    - You may want to use some even higher lever wrappers like Keras, but I wouldn't cover them during the class
  - Jupyter
    - A nice interactive development framework
  - Tensorboard
    - If you want to see what's going on in your network

# Baby TensorFlow
## - Computational Graphs

- In TF things are represented with computational graphs
  - That consist of tensors and operations on them
- Building and running the computational graphs are separate
  - You define the graph first
    - This procedure is to represent what you want to do symbolically
    - It doesn't do any computation
  - Running the graph
    - Visit the graph nodes as required by the session's **run** method
    - Actually does the computation
- What for?
  - One obvious reason is to make your life easier by doing differentiation for you
    - So, it does the differentiation on the symbolic representations
  - And then actually calculate the gradient when you run the graph

# Baby TensorFlow
## - Tensors

- Three different kinds of tensors
  - `tf.Variable`: something TF can change
  - `tf.placeholder`: something you can change
  - `tf.constant`: something nobody doesn't change once initialized
- For example,
  - You want to train a classifier from a very large training dataset $\boldsymbol{Y} \approx \mathcal{G}(\boldsymbol{X}; \mathbb{W})$

This will be a very large data matrix

$$\frac{\partial \mathcal{E}}{\partial \boldsymbol{W}^{(2)}} = (\hat{\boldsymbol{Y}} - \boldsymbol{Y})\boldsymbol{H}^\top$$

A heavy matrix operation, maybe too large for the main memory

  - So, you want to do something called **Stochastic** Gradient Descent (SGD), where
    - You sample a data point and calculate the gradient by using it
    - Or, you divide the entire data matrix into smaller pieces, called **minibatches** and calculate gradients for each of them
  - Then,
    - `tf.Variable`: $\boldsymbol{W}^{(2)}$ ← You can set TF to update them during the run (of course you can initialize them as you want)
    - `tf.placeholder`: $\boldsymbol{H}_{:,t}, \boldsymbol{Y}$ ← You need to feed this SGD samples or minibatches at every iteration
    - `tf.constant`: vector of ones for bias ← Once initialized nobody cares about this

# Baby TensorFlow

## - Running the graph

- If everything is symbolic, when do we actually do the operation?
  - In TF, there's a concept called "session" where all the variables and operations are actually taken care of
  - `tf.session.run(fetches, feed_dict=None, ...)`
    - `fetches`: The graph element you want to run
    - `feed_dict`: You can assign some values to the `placeholder` tensors
- Chain reaction
  - Running a graph element means running all the operations and evaluating tensors that are necessary for the fetched one
- For example,
  - You are interested in the training cost $\mathcal{E} = -\sum_t \sum_c \boldsymbol{Y}_{c,t} \log \hat{\boldsymbol{Y}}_{c,t}$
  - You define this cost as a graph element
  - If you do `sess.run()` on it, you need to know $\hat{\boldsymbol{Y}}$, the prediction of the class labels
  - `sess.run()` traverses the graph to get this done
- `tf.gradients(cost, variables)`
  - Does the differentiation w.r.t. the variables
  - Most of the time you'll use a wrapper that calls this function internally

# Baby PyTorch

## - Difference between TF and PyTorch

- The look
  - TF: you construct the graph first and then run the graph
  - PT: it does construct the graph but things are more seamless
    - Therefore the source code is more similar to the regular numpy codes
- The tensors
  - TF: there are three different types of tensors
  - PT: a tensor is basically an N-dimensional array that has nothing to do with deep learning
    - But you can specify a tensor in the GPU memory if you want
    - `torch.autograd.Variable` defines a wrapper that turns a tensor into a PT variable
      - With which you can do all the cool things like automatic gradient calculation
- GPU computing
  - TF: if the same operation has two kernels for both CPU and GPU computing, GPU version gets the priority
  - PT: there are some predefined GPU data types and a way to convert tensors
    - e.g. `torch.FloatTensor` versus `torch.`**`cuda`**`.FloatTensor`
    - `cuda()`: copies the tensor in the GPU memory

# Baby PyTorch

## - Difference between TF and PyTorch

- The gradients
  - TF: `tf.gradients(cost, variables)` differentiates the cost function w.r.t. the variable during the graph construction
    - The procedure is symbolic, so are the derivatives
  - PT: all variables have their own `.grad` component that holds the actual gradient values
    - It's called "autograd" so things are still automatic, but treated differently
    - When `.backward()` method is called, the node is differentiated w.r.t. the leaf nodes in a recursive way
    - To do so, the forward pass records the input to the node
    - The backward pass calculate the gradients of all the intermediate nodes on the way back

# TF versus PT

- TensorFlow
  - Google
  - Seems to have a larger user community
  - Said to be better in building a serious project
- PyTorch
  - Facebook
  - Rapidly evolving
  - Easier to quickly see the proof of concept

I do have my own preference, but for this course choose whatever you prefer

# Thank You!

**Minje Kim**
Department of Intelligent Systems Engineering
Email: minje@indiana.edu
Website: http://minjekim.com
Research Group: http://saige.sice.indiana.edu
Meeting Request: http://doodle.com/minje