ML@AIMAS Summer School
Utilizarea platformelor de Deep Learning

PyTorch Intro: Tensors, Loading, Building, Training and Inference for MLP

Dan M. Novischi

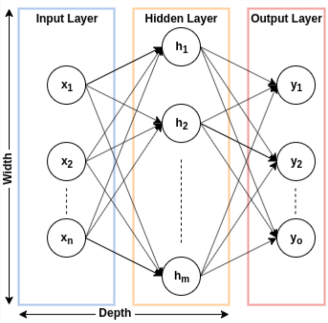
dan_marius.novischi@upb.ro

University POLITEHNICA of Bucharest
Faculty of Automatic Control And Computers

MLP/Feedforward NN Quick Reminder

- Neural networks are composed of a base unit called **neurons**.
- Neuron units are stacked together to form a **layer**.
- Layers are organized in a chained fashion to form a **network**:
  - **input layer** is the actual input feed into the network.
  - **hidden layer** represents interior layers.
  - **output layer** gives the network target predictions.
- This type of network is called a **Fully Connected Neural Network** or **Multilayer Perceptrons**.

MLP/Feedforward NN Quick Reminder



- The **architecture** of the network is described by two parameters, namely:
    - **depth** which represents how many layers are within the network, i.e. how deep the network is.
    - **width** which represent how many units are in the hidden layers.
- Later on, you will see that layer type is a another parameter.

What is Pytorch?

- A thin framework over python for building Deep Nets.
- Very pythonic, you basically code as you usually do.
- Dynamically generates neural network computational graphs.
- It’s object oriented with powerful debugging support.
- As fast as other frameworks (e.g. TensorFlow, Keras, CNTK)

What are Pytorch main componets?

- torch.nn - modules and extensible classes for building NNs.
- torch.autograd - differentiable tensor operations support.
- torch.nn.functional - loss functions, activation functions, convolution operations...
- torch.optim - optimizers for used for training SGD, AdaGrad, RMSProp, Adam ...
- torch.utils utilities for data sets and data loaders.
- Plus many other useful extensions: torchvision, torchaudio, torchtext...

What is a Tensor?

- Up to now we have used concepts such as numbers, arrays, matrices, etc... to describe concepts of both mathematics, physics and computer science.
- For example, the table below shows the mapping between concepts we used interchangeably between mathematics and computer science.

Computer Science	**Mathematics**
number	scalar
array	vector
2d-array	matrix

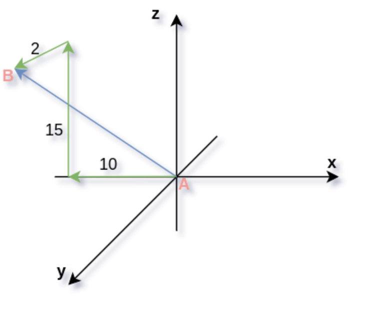
- These represent **specific instances** of a more **general concept** called **tensor**.
- For physics people, checkout [1]

Zero-basis Tensor Example

- While in computer science we use **tensor** to mean **nd-array**, the tensor generalization has a deeper meaning, i.e. the notion is accompanied certain proprieties and transformations.
- Spouse, you what to specify the temperature in a room.
    - We need a scale like Kelvin, Celsius or Fahrenheit.
    - Based on the chosen scale we specify a magnitude, i.e. a number that specifies the temperature.
- Hence, we need a single magnitude (number) to specify the temperature.
- Since the temperature is just a magnitude, it has no direction.
- In other words, we have zero-basis vectors per component to specify the temperature.

One-basis Tensor Example

- Suppose you want to specify the displacement between points A and B in 3 dimensions.
  - We need to know the distance (magnitude) $\left\|\vec{AB}\right\|$ between A and B.
  - We also need the direction of vector $\vec{AB}$.



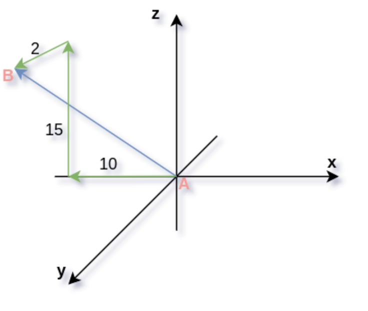
Displacement 3d-vector example.

One-basis Tensor Example

- We could write the vector $\vec{AB}$ in term of its components.

$$\vec{AB} = -10\vec{i} + 2\vec{j} + 15\vec{k}$$

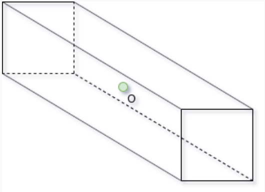
- Note that we need three components: $10\vec{i}$, $2\vec{j}$ and $15\vec{k}$, where each component requires one-basis vector to specify $\vec{AB}$.



Displacement 3d-vector example.

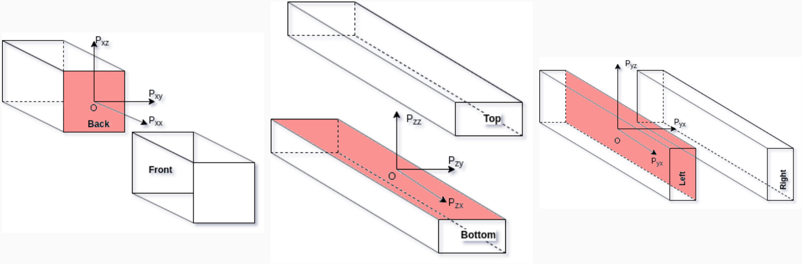
Two-dimensional Tensor Example

- Consider a roof support beam in a house as depicted below.
- We want to represent the force acting at point O.
- The force can be decomposed along 3 surfaces.



Household roof beam.

Two-dimensional Tensor Example



Beam stress tensor graphic decomposition.

- P_{ab} specifies a component single component along a surface and a direction in which the force is acting on that surface.
- $a = \{x, y, z\}$ specifies the surface $\perp$ direction.
- $b = \{x, y, z\}$ specifies the direction of the force component.

Two-dimensional Tensor Example (Cont'd)

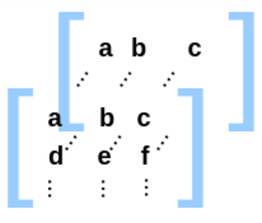
- Note, we have a total of 9 components, where each component has a magnitude and two basis vectors:
    - One-basis vector for the cross-section area a.
    - One-basis vector for the direction of the force given a cross-section b.
- Hence we have two basis vectors per component corresponding to a and b.
- We can encode this in:

$$P = \begin{bmatrix} P_{xx} & P_{xy} & P_{xz} \\ P_{yx} & P_{yy} & P_{yz} \\ P_{zx} & P_{zy} & P_{zz} \end{bmatrix}$$

  For example P_{yx} has one-basis vector for the cross-section in y direction and one-basis vector for the direction of the force in x direction.

Multi-basis Tensor

- In n-dimensional space a tensor is an (mathematical) object that has m-indices.
- The number m of indices is given by the **number of basis vectors** required to (fully) specify a component of the tensor. This is called the **tensor rank**.
- Temperature has rank 0, $\vec{AB}$ has rank 1 and P has rank 2 and all are specific instances of tensors.



Rank 3 tensor example.

Tensor Rank

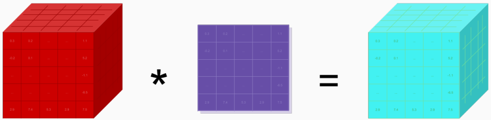
- Note, that we can represent tensors using nd-arrays. However tensors are more than nd-arrays.
- For example, $\vec{AB}$ has a dimension of 3, i.e. 3 components in the 3D Euclidean vector space.
- However, a rank 3 tensor (or a tensor in 3D) can have many more components. The rank 2 stress force P has 9 components.
- **To remember:** In computer science the rank of a tensor tells us how many indexes we need to access a specific data element and vice-versa.
- Also note that a tensor can have a different number of components on each dimension.

Tensor Axis

- Having a rank *n* tensor, also means that the tensor has *n* axes.
- For instance, a rank 2 tensor has 2 axes.
- The **length of an axis** repr. the **number of components in that axis**.
- The product of axis lengths within a tensor gives us the total number components of the tensor.
- So, for a rank 2 tensor with the first axis length of 3 and the second with a length of 4, the available indexes on each axis are:
  - $i = \overline{0,\ 2}$ or $i = \overline{1,\ 3}$
  - $j = \overline{0,\ 3}$ or $j = \overline{1,\ 4}$

Tensor Shape

- Generally, all the rank/axis information is encoded in a tensor's shape.
- The **number of components in the shape** of tensor specifies its **rank** and therefore the number of axes.
- The **values of the shape components** represent each of the axes lengths.
- The shape helps us visualize multi-dimensional tensor operations as the one below.



Typical tensor operation visual representation within a network.

References I

K. Dullemond and K. Peeters.
Introduction to tensor calculus.
Kees Dullemond and Kasper Peeters, 1991.