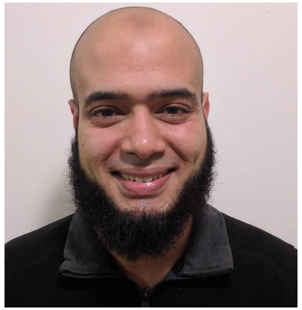
# Machine Learning

# Debugging DNNs

**Mostafa S. Ibrahim**
*Teaching, Training and Coaching for more than a decade!*

*Artificial Intelligence & Computer Vision Researcher*
*PhD* from Simon Fraser University - Canada
*Bachelor / MSc* from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



## Tips for Deeping Learning Projects

- Whenever you can start from a running baseline code, just do it
  - This will speed your progress
  - Add one change at a time and make sure things are improving (or at least not worse)
- If you can start from a pre-trained model, go with it
- Starting from scratch can be challenging.
  - Always start with the simplest baseline first. Easier to code and debug
- Do a lot of data verification and visualization
  - Allow small data mode for easier debugging
  - Always log enough details about the model
- Start with a simpler data
  - For example, you may start from easy classes and avoid some imbalance classes
- Visualize your different curves, LR curve and data samples

## Dataset issues

- If you are using a pretrained model, you should **normalize** your data according to it: E.g. [0-1] range? Specifica mean/std normalization?
- Print the input of the first layer and make sure it works well
- In your tensorboard, you may save input data/images and make sure they are as expected (For example, **augmentation effect**)
- Make sure labels are correct and data is shuffled
- If you have multiple datasets, try them independently
    - For your classifier, train on standard datasets, e.g. CIFAR10
- Make sure no data leakage issues

## Sanity Check: Overfitting a Single Example

- Sometimes something wrong from the network to the loss, as a result some weights are actually still random and not updated
- A simple trick here is to create a batch of a SINGLE example
- If your training managed to train well on it / test well, then this means parameters are really updated
  - You may also compare weights before and after

## Your network

- Complex models requires a lot of data
  - Or start from a pretrained model with fair amount of data
  - Try with or without freezing some layers
- Start with promising setups
  - AdamW optimizer with 0.001 LR
  - Reasonable batch size such as 64 / 128
  - Simple StepLR scheduler and monitor the loss
  - Use popular deep learning initializers / use default initializers
- Explore the effect of data augmentations individually
  - Find the right hyperparameters. Guide yourself with popular choices
- Combining different losses? Make them on same magnitude + weight factors
- Check for hidden dimension errors
  - Use different numbers (or prime numbers to verify matchings)

## Training

- Features scaling
- Explore different hyperparameters, one at a time
  - Start from common ones. Review recent papers for relevant problems
- Review your regularization techniques
  - This can play a role in overfitting / underfitting
  - Dropout in the wrong location / Use Batchnorm
- Avoid early stopping / late stopping
- Explore different optimizers / learning rates
- NANs
  - Input is NAN / Code bug
  - High learning rates (try AdamW) / review scheduler
  - Division by zero (For example the derivative of sqrt is 1/[2sqrt(x)]

## Relevant Materials

- 37 Reasons why your Neural Network is not working

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."