JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

EN.601.482/682 Deep Learning

It’s not working! Help!

Mathias Unberath, PhD
Assistant Professor
Dept of Computer Science
Johns Hopkins University

Training DL Models Fails Silently

- Misconfiguring code → Error message
- Bad training setup → No obvious feedback to identify problems

→ **Systematic way to identify and fix bugs**

Know Your Data

- Visualize examples to gain understanding of the distribution
- Is there imbalance / bias / heavy outliers? Think about quantifying this!
- Understand the challenges: Local or global problem?
- Pre-processing to remove unnecessary sources of variation?

Get Baselines

- Gain trust in training and evaluation skeleton code
- Pick the simplest model, make sure it's correct, and test the setup
- Things to test
  - Verify loss at initialization
  - Verify input to the model (pre-processing pipeline!)
  - Verify decreasing loss (during optimization and when using a model with higher capacity!)

Overfit & Regularize

- First, find a model with capacity high enough to overfit the data
  - Don’t be a hero: Use models that work! … at least in the beginning
  - Add complexity successively (not all at once!)

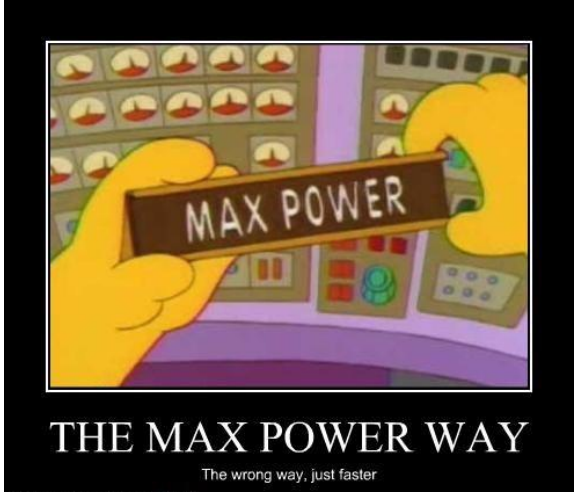
- Second, regularize the model to improve generalizability
  - Get more data, if necessary, via augmentation
  - Pre-training and transfer learning
  - Conventional regularization (batch norm, dropout, weight decay, early stopping)

- Finally, verify you have a proper classifier
  - Visualize filter kernels and responses of early layers
  - Consider class activation mappings and similar techniques (later lecture)

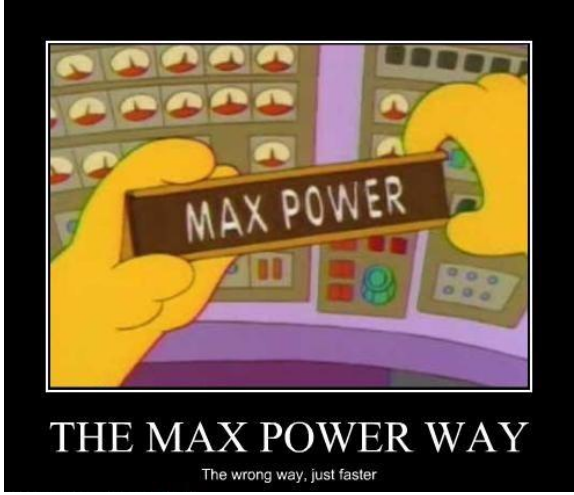
4 Training Setups

1. The Max Power Way
2. DICE DICE Baby
3. Randomization TO THE MAX
4. My test performance? It’s over 9000

The Max Power Way



The Max Power Way



- What you should have observed
  - Accuracy is poor
  - Loss curve does not look right (you are getting NANs?)
- Problem: Learning rate is too high, and optimization diverges
- Solutions
  - Either decrease learning rate (~0.01)
  - Modify the architecture (batch norm makes optimization more stable!)

DICE DICE Baby

DICE DICE Baby

- Numpy
  - DSC_num = 2*(A*B).sum()
  - DSC_den = (A*A+B*B).sum()
  - DSC = DSC_num(A,B) / DSC_den(A,B)

- Pytorch
  - DSC_num = 2*(A*B).sum(dim=1)
  - DSC_den = (A*A+B*B).sum(dim=1)
  - DSC = (DSC_num(A,B) / DSC_den(A,B)).mean()

Randomization TO THE MAX

Randomization TO THE MAX

- What you should have observed
  - Accuracy is poor
  - Loss curve does not look right

- Problem: Independent random shuffle of instances and labels

- Solution: Do not shuffle independently
- How to spot this?
  - Visualize your data and labels
  - Gain confidence in your skeleton before adding complexity!

My test performance? It’s over 9000

My test performance? It's over 9000

- In the very first epoch, you achieve 100% testing accuracy…
  … which is different from your training accuracy

- Concatenation of 8 and 0s, but then there's a 90-10 split
- It's a binary problem (0/1 type labeling): What's the expected value after init?

- Solution: Randomly shuffle instances (while keeping the label assignments)