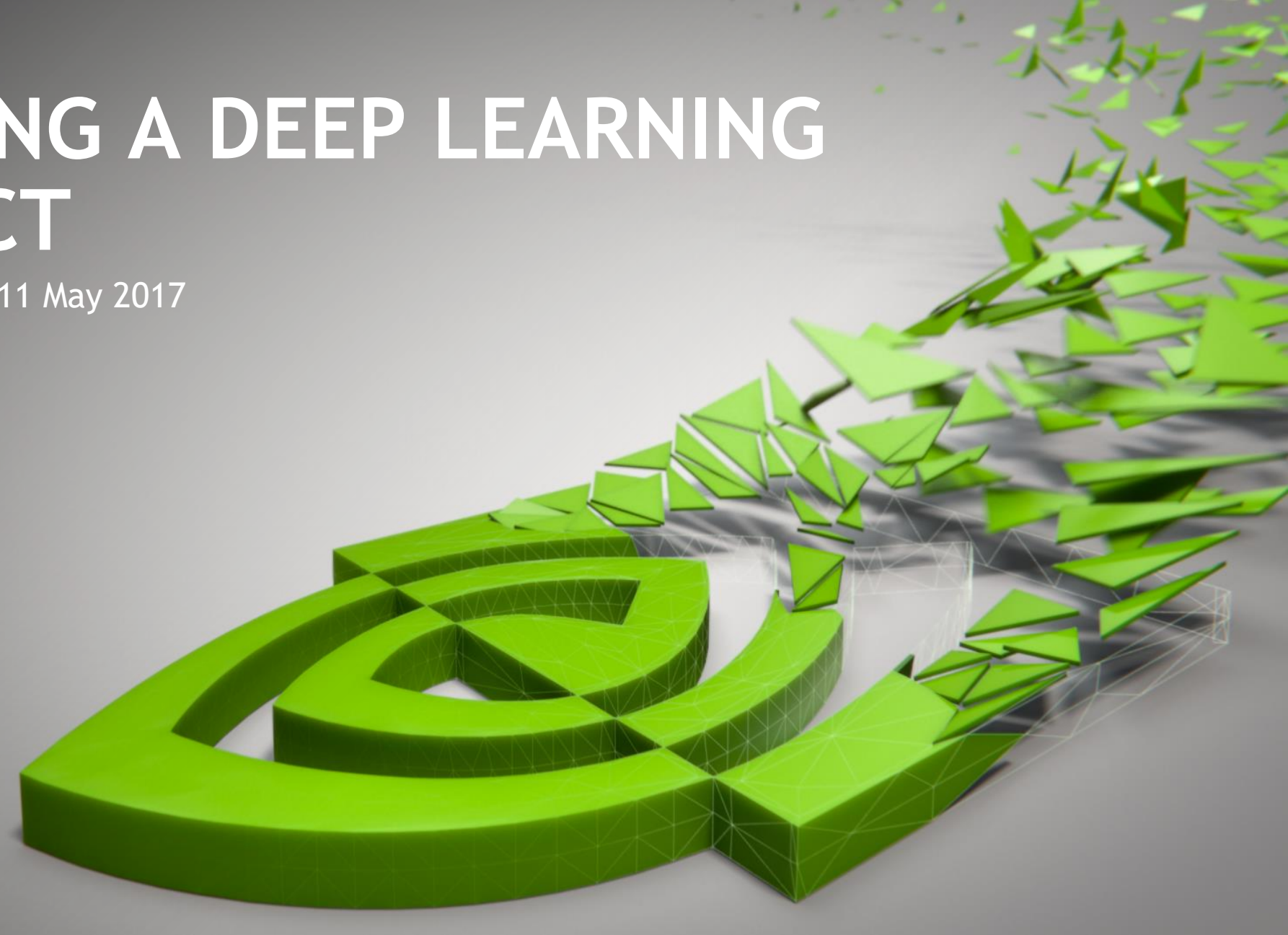
STARTING A DEEP LEARNING PROJECT

Bryan Catanzaro, 11 May 2017

NVIDIA



Supervised learning
(learning from tagged data)

X → Y

Input Image → Output tag: Yes/No (Is it a coffee mug?)

Data:

 → Yes

 → No

Learning X → Y mappings is hugely useful

EXAMPLE X->Y MAPPINGS

Image classification

Speech recognition

Speech synthesis

Recommendation systems

Natural language understanding

Most surprisingly: these mappings can generalize

DEEP NEURAL NET

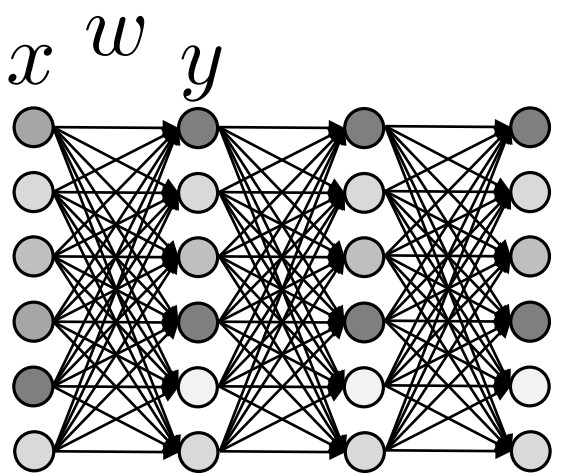
A very simple universal approximator

$$y_j = f\left(\sum_i w_{ij} x_i\right)$$

One layer

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

nonlinearity



Deep Neural Net

WHY DEEP LEARNING

Scale Matters

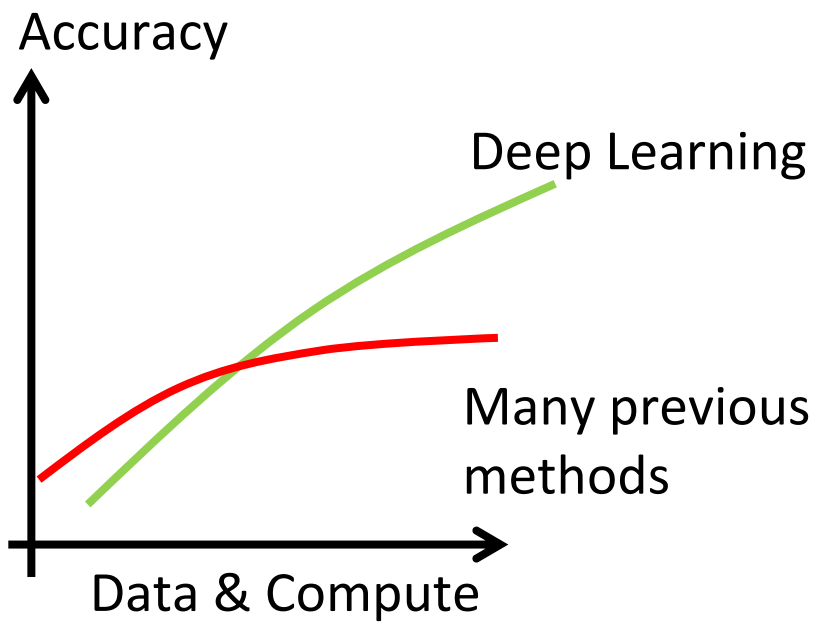
Millions to Billions of parameters

Data Matters

Regularize using more data

Productivity Matters

It's simple, so we can make tools



Deep learning is most useful for large problems

SUCCESSFUL DEEP LEARNING

What characteristics do successful deep learning applications share?

How to prepare to use deep learning?

1. DATASET

Deep learning requires large datasets

- Without a large dataset, deep learning isn't likely to succeed
- What is large? (typically thousands to millions)

Labels are a huge hassle

- Getting someone to decide the "right" answer can be hard
- If a dataset requires skilled labor to produce labels, this limits scale

2. REUSE

Making deep neural networks is expensive

- Computation
- Data acquisition
- Engineering time

So deep learning makes sense if a model can be reused

If small changes to the problem invalidate the model, it's not a good fit

- For example, if a model has to be retrained for each level of a videogame, this makes it hard to deploy

3. FEASIBILITY

Can you describe the problem as an X -> Y mapping?

Speech recognition

Image classification

Or does it require “strong AI”

“Magic goes here”

What level of accuracy is required for the application to succeed?

4. PAYOFF

Generally needs a big payoff to justify investment

If you had an oracle for this problem, what would change?

What is the speed of light opportunity?

Self-driving cars - $T market opportunity

Cafeteria menu predictor - ???

5. FAULT TOLERANCE

Every statistical method fails at times

Plan for occasional failure:

- Guard rails
- Heuristics

All models are wrong, but some are useful -- George Box

TRAINING, VALIDATION, TEST SET

Training set:

bang on this data all you want

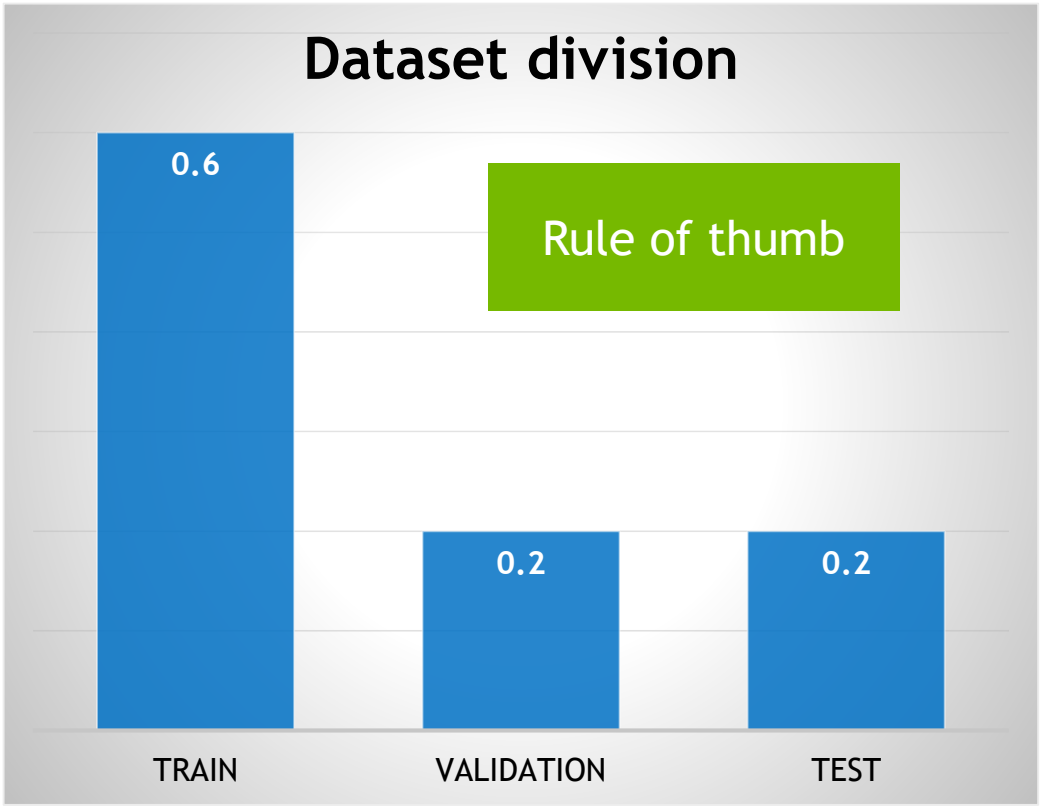
Validation set:

periodically during training, check

(are we overfitting?)

Test set:

rarely (weekly), evaluate progress



OVERFITTING

Neural networks can memorize details of training set

This can lead to loss of generalization

In other words: failure

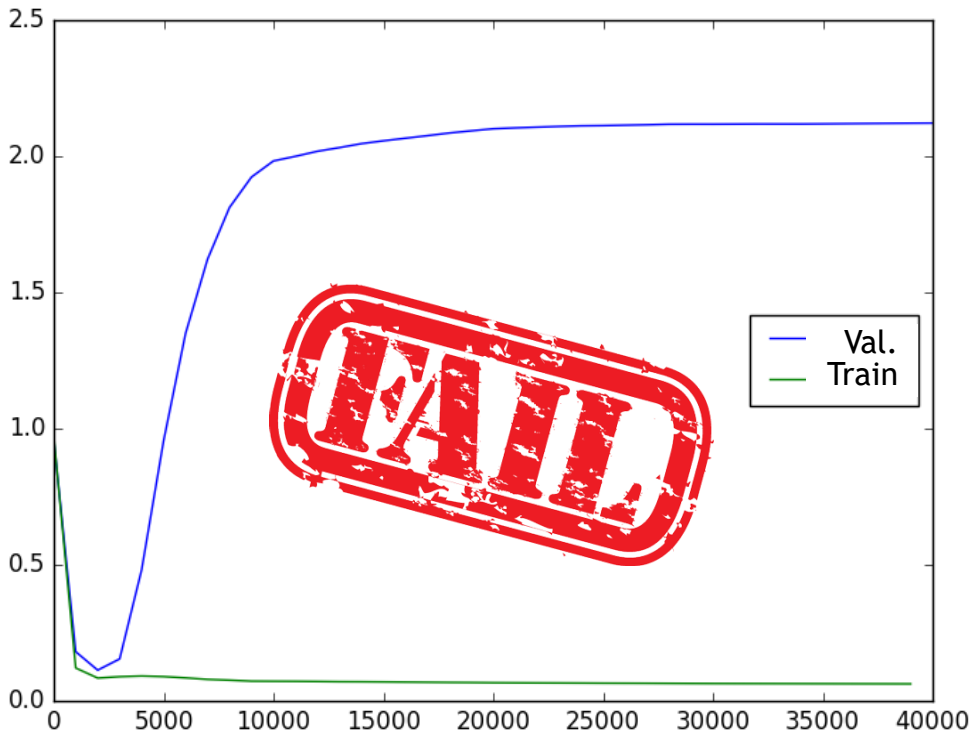
It often looks like this:

Training loss goes down

Validation loss goes up

Your network is probably too big

Or your data is too small



@ctnzr

MAKING YOUR TEST SET

Garbage in, garbage out

Many choices while partitioning dataset into train, validation, test

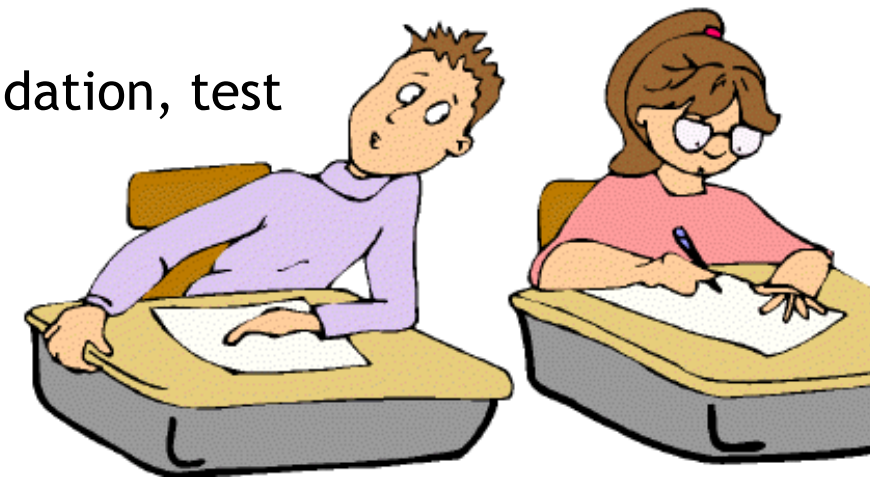
Critical to do this right

Training set should be representative of testing set

But cannot include the testing set

If you don’t set up your test set to prove generalization

You will get overfitting



@ctnzr

THE EXTERNAL TRAINING LOOP

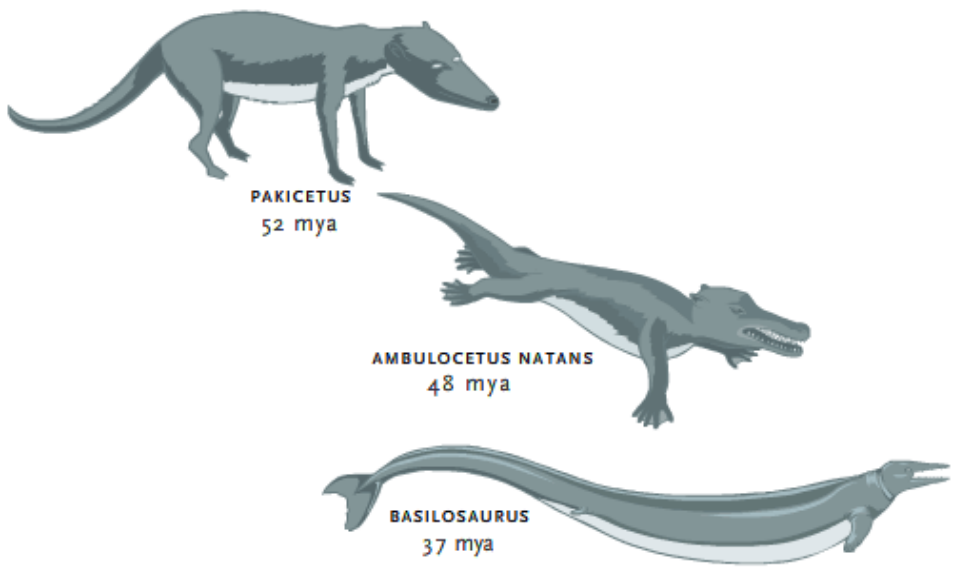
What happens if you peek at your test set too often?

- Survival of the fittest
- Evolution
- Overfitting, like it or not

This is why competitions have rules

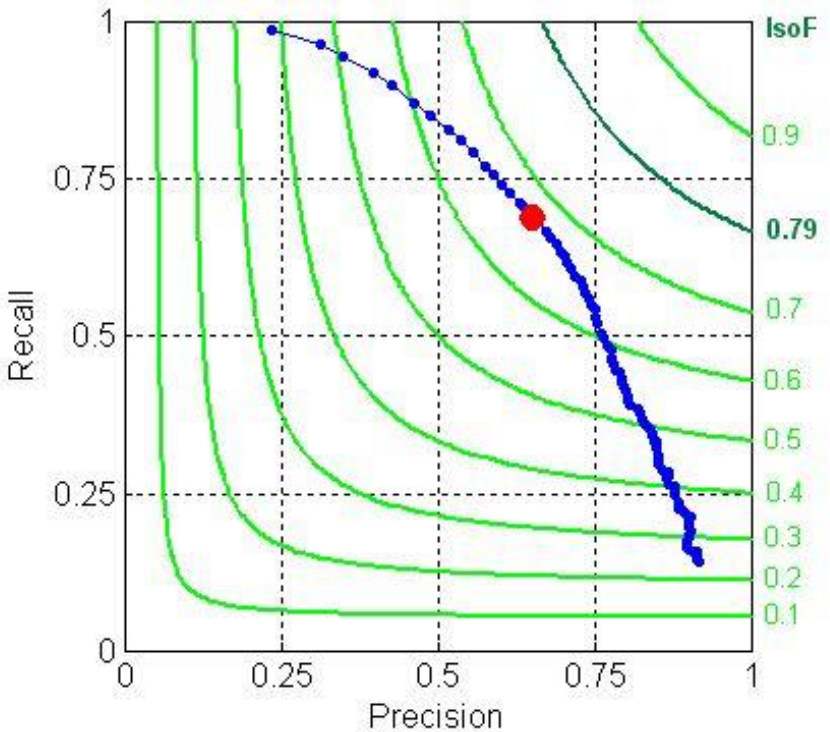
- Can’t test your model too often

Hierarchy of test sets



PRECISION & RECALL

For binary classifier



Precision: when you said you found it, how often were you right?

Recall: what percentage of true things did you find?

Fundamental tradeoff here:

Only care about precision: always say no

Only care about recall: always say yes

Area under the curve

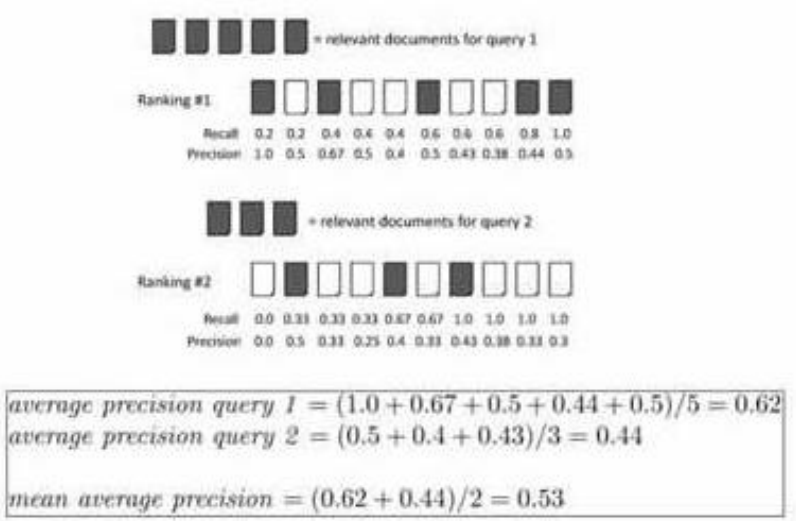
ACCURACY

Before starting a project, you should figure out what success looks like

This can be surprisingly hard to pin down

Lots of ways to measure it: Area Under Curve, specificity/sensitivity, mean average precision

First thing to do: get a test set, figure out how to measure accuracy



CAN SOMETHING SIMPLER WORK?

After you have a test set and an accuracy metric

You should try a very simple model (linear regression, logistic regression, random forest)

This gives you a baseline on which to improve

If the simple thing is already good enough, you've won!

DATA CULTURE

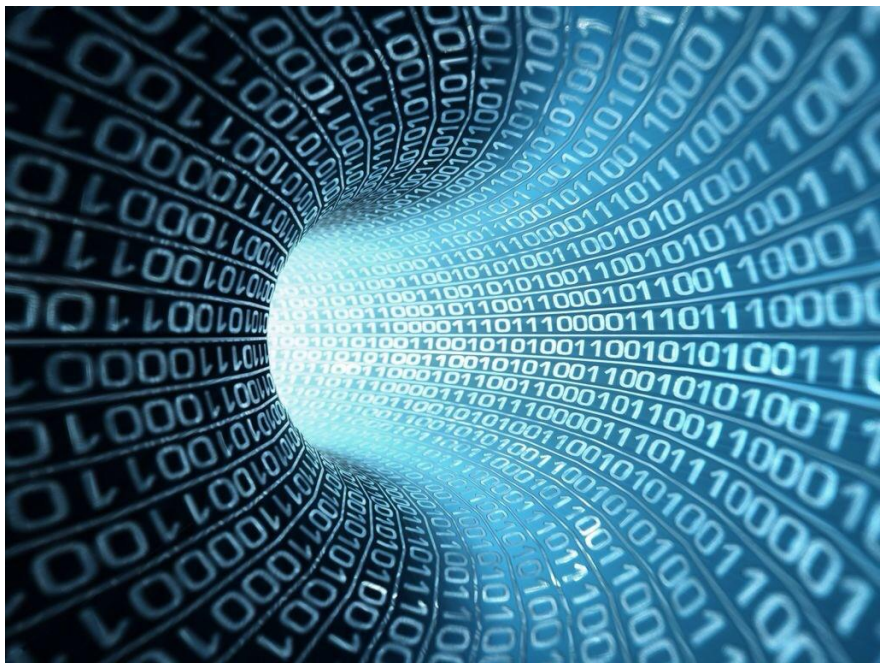
Often, data is undervalued

We need to preserve as much data as possible

Years down the road, it could be useful

All of us should think of ways of building up data

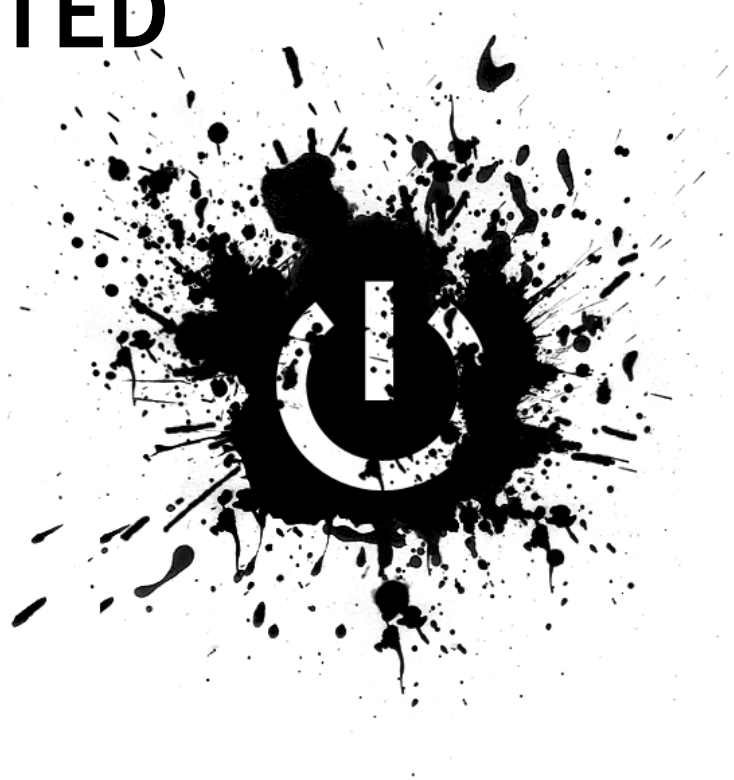
Labels are especially useful (like feedback, or sorting, etc.)



Would be great for Nvidia to have centralized data stores

So others could experiment

HOW DO I GET STARTED



Take a machine learning class! (DLI)

Learn a framework: Tensorflow, Torch, Caffe, CNTK, Mxnet, Keras, Theano

Brainstorm useful X-Y mappings

Bias towards action: experiment! Try it out!

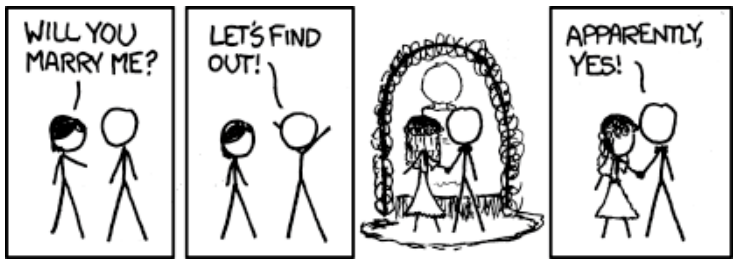
BIAS TOWARDS EXPERIMENTATION

Deep Learning is an empirical field

It’s hard to know whether an idea will work

Some, surprisingly, do work

Some, surprisingly, don’t



If you have convinced yourself you’ve framed the problem appropriately

You should then start trying things out

CONCLUSION

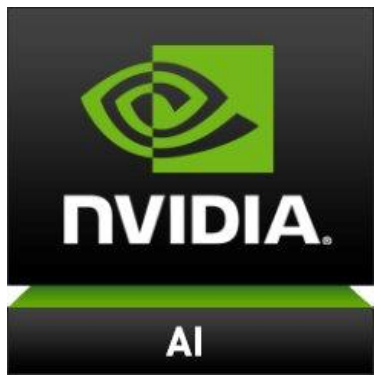
We’re all excited about Deep Learning

As you think about your own DL applications, consider:

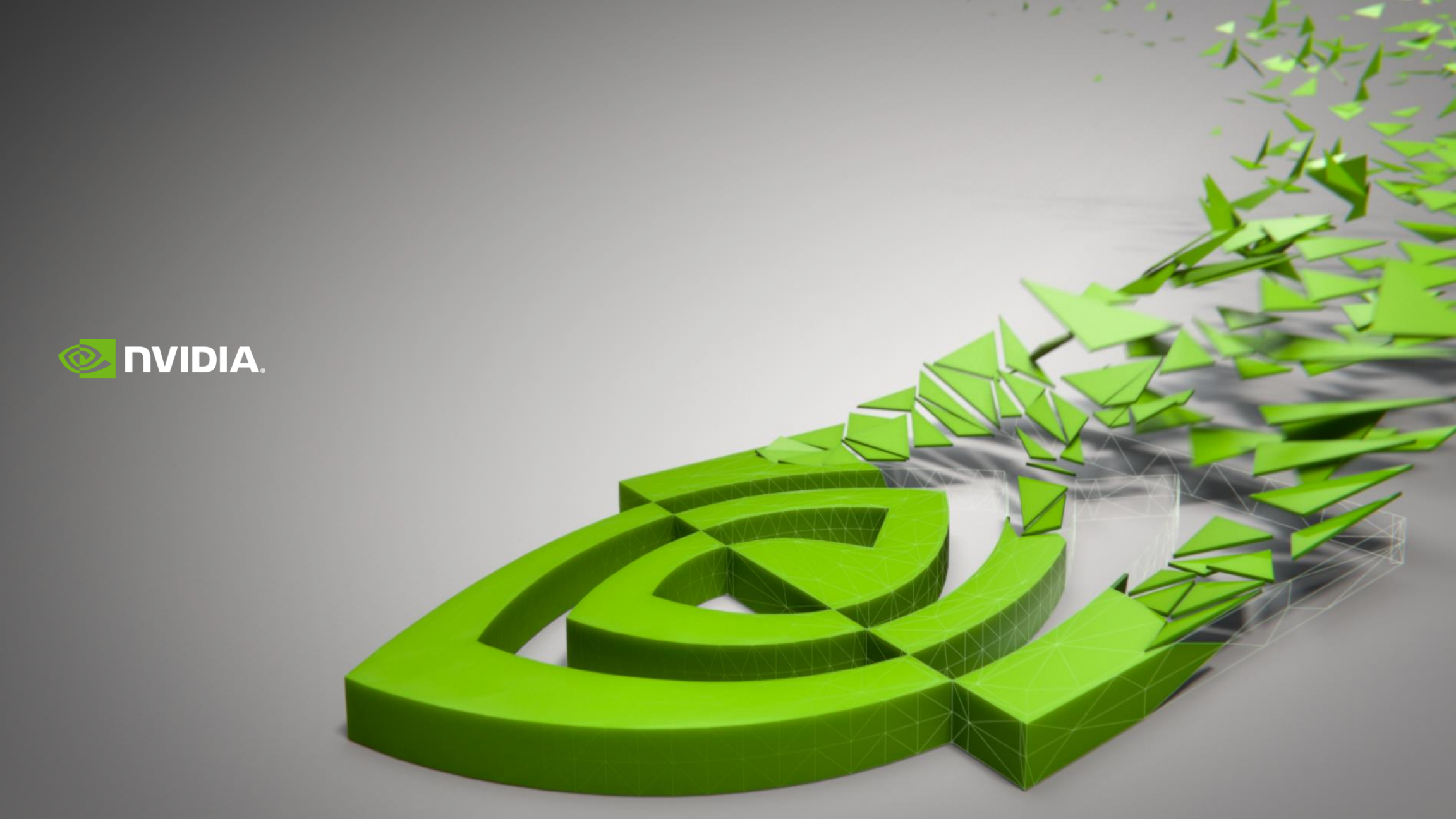
1. Dataset
2. Reuse
3. Feasibility
4. Payoff
5. Fault Tolerance

Make a test set, figure out how to measure accuracy

Experiment! Try it out!



@ctnzr


NVIDIA