Deep Learning in Practice

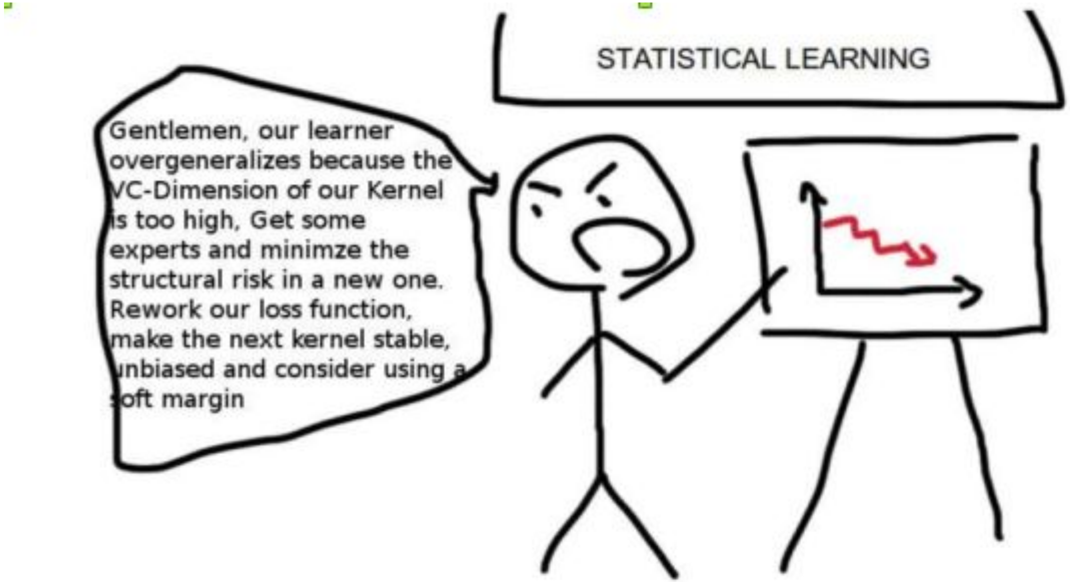
Dewang Sultania

Disclaimer

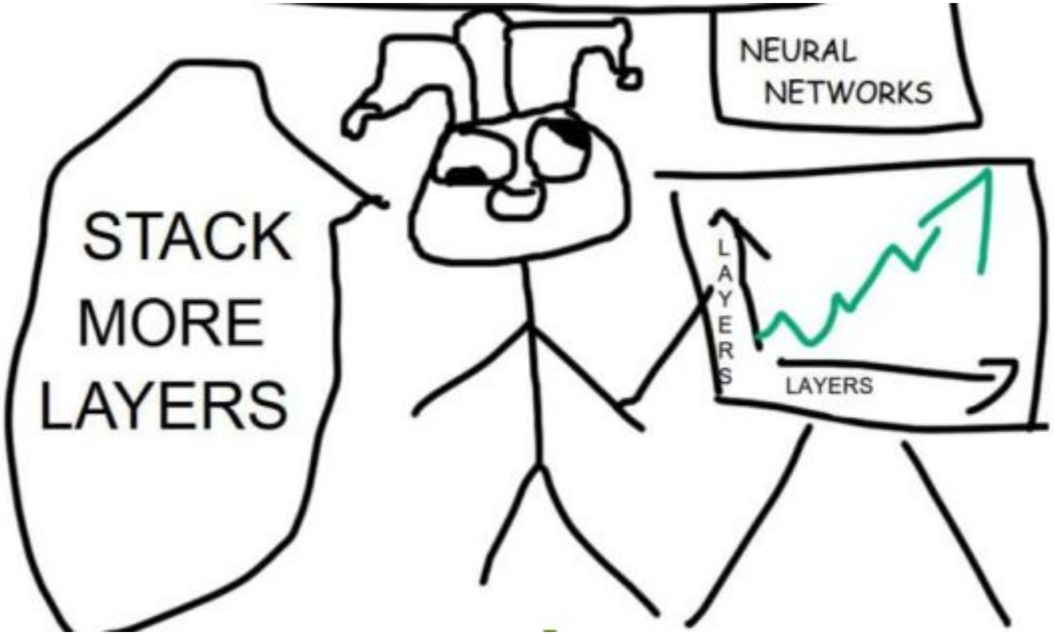
The ideas discussed here are a cumulation of ideas from various industry experts including but not limited to Andrew Ng, Andrej Karpathy, Jeremy Howard, Pieter Abbeel, Sergey Karayev, Josh Tobin etc. I do not claim to have any theoretical justification for these ideas, these are the things that they have found to work when deploying machine learning applications at large scale and I found it worth sharing.

Some slides have been adapted from Prof Konrad Kording's CIS 700: Deep Learning from Data Science course

Steps of an ML application



Steps of a DL application



The End

Topics for Discussion

- How neural networks fit in the context of a bigger product.
- Debugging machine learning projects.
- Failure modes in Deep Learning.

Toy Example

(x, y, z) **Position (L2 loss)**

(ϕ, θ, ψ) **Orientation (L2 loss)**

Xiang, Yu, et al. "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes." arXiv preprint arXiv:1711.00199 (2017).

Steps of a ML application

Planning & project setup

- Decide to work on pose estimation
- Determine requirements & goals
- Allocate resources
- Etc.

Steps of a ML application

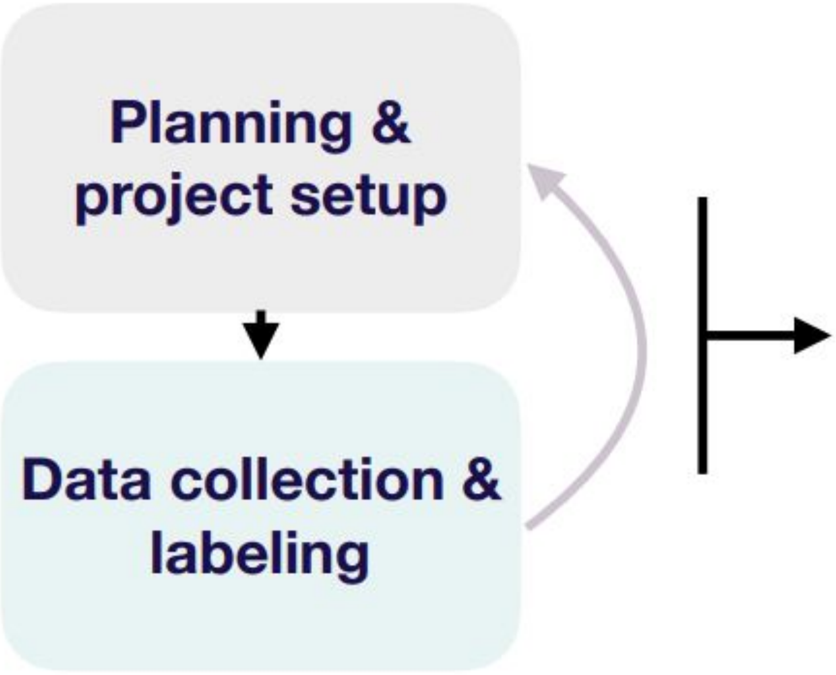
Planning & project setup

↓

Data collection & labeling

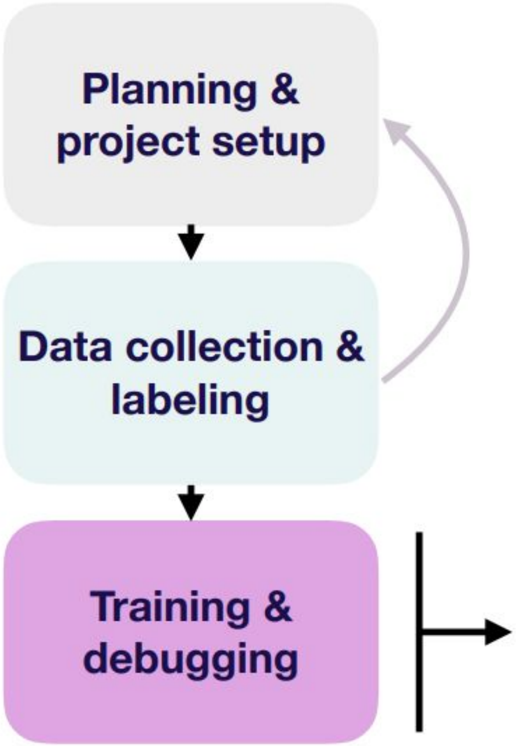
- Collect training objects
- Set up sensors (e.g., cameras)
- Capture images of objects
- Annotate with ground truth (how?)

Steps of a ML application



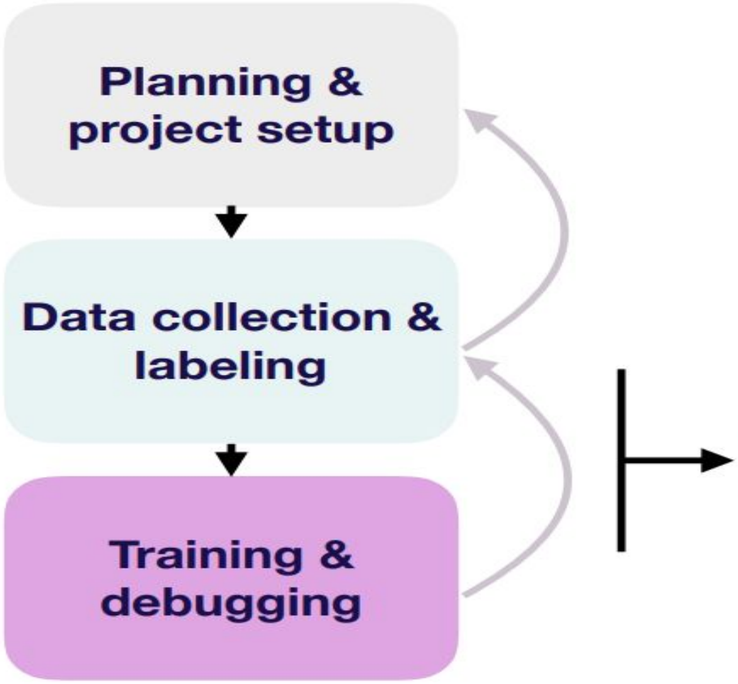
- Too hard to get data.
- Easier to label a different task (e.g., hard to annotate pose, easier to annotate per-pixel segmentation)

Steps of a ML application



- Implement baseline in OpenCV
- Find SoTA model & reproduce
- Debug our implementation
- Improve model for our task

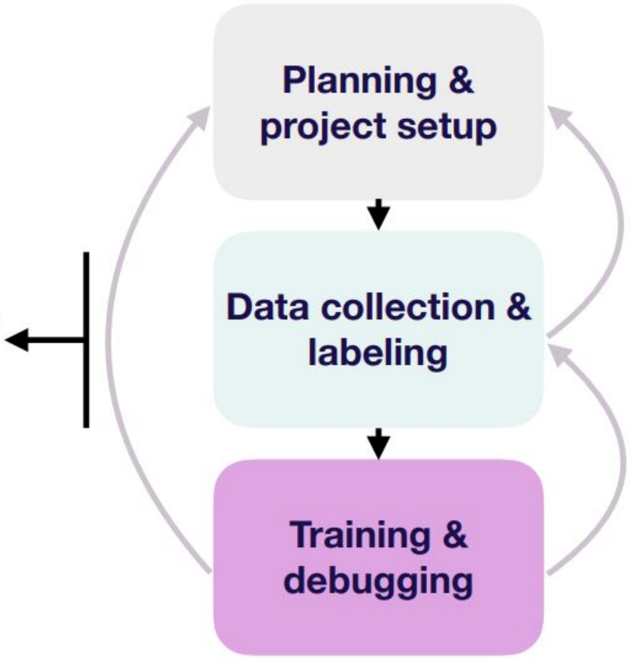
Steps of a ML application



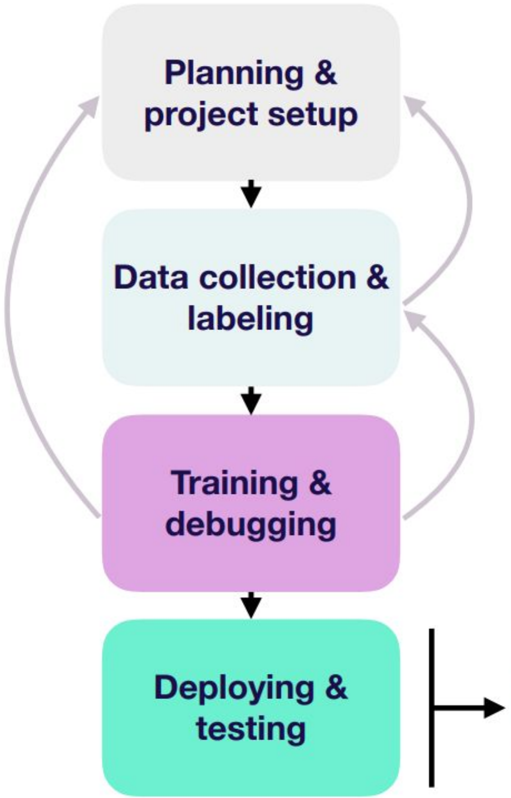
- Collect more data
- Realize data labeling is unreliable

Steps of a ML application

- Realize task is too hard
- Requirements trade off with each other - revisit which are most important

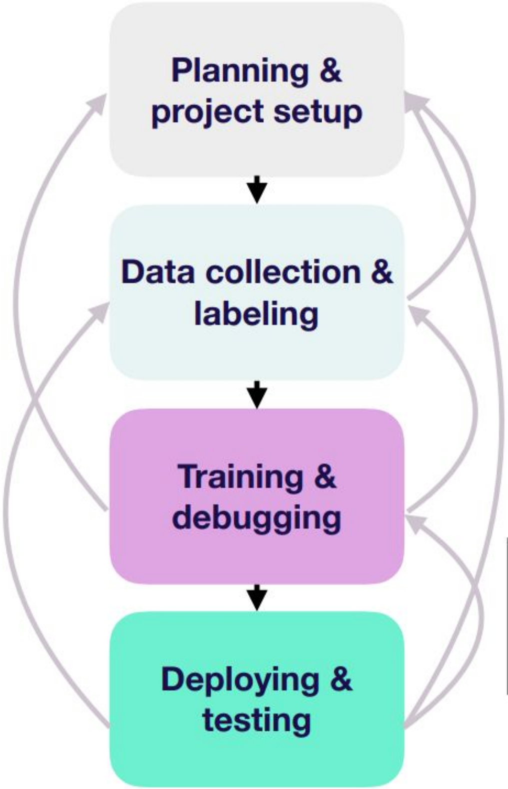


Steps of a ML application



- Pilot in grasping system in the lab
- Write tests to prevent regressions
- Roll out in production

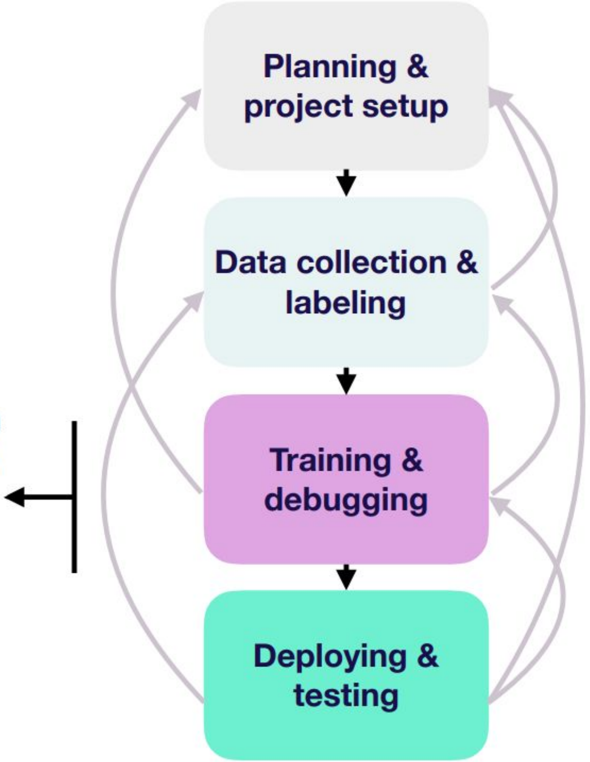
Steps of a ML application



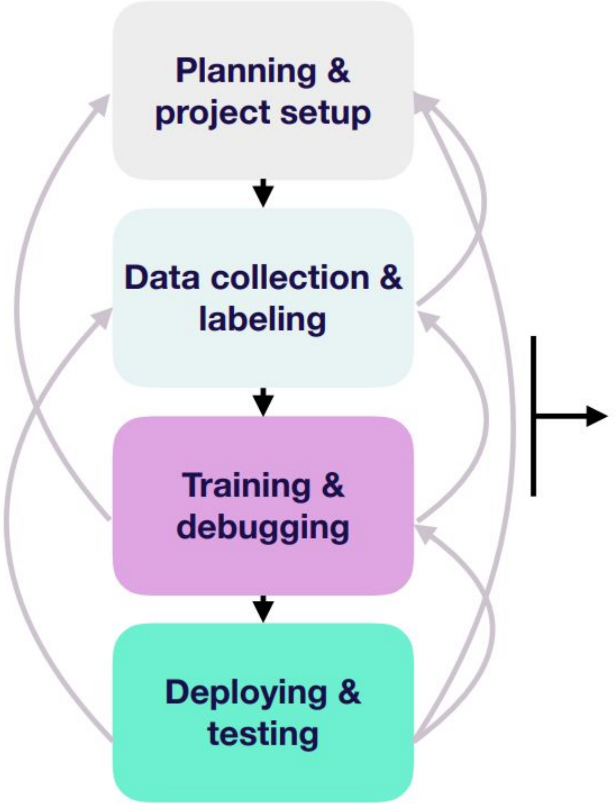
- Doesn't work in the lab - keep improving accuracy of model

Steps of a ML application

- Fix data mismatch between training data and data seen in deployment
- Collect more data
- Mine hard cases



Steps of a ML application



- The metric you picked doesn't actually drive downstream user behavior. Revisit the metric.
- Performance in the real world isn't great - revisit requirements (e.g., do we need to be faster or more accurate?

Debugging Neural Networks

Error Analysis

- Manually examining the mistakes algorithm is making can give you insights into what has to be done next
- Look at your validation set to evaluate ideas

As an example let's say you are building a cat classifier app and you get 90% accuracy on your validation set, equivalently 10% error.

Error Analysis

So now let's say you find that your algorithm is misclassifying some of the dogs as cats.

The question to ask yourself is should you try to make your cat classifier do better on dogs?

Error Analysis

- Get ~100 mislabelled dev set examples
- Count up how many are dogs.

Suppose out of the 100 sampled examples 5 are dogs, so the best you can hope to achieve in this case is 10% error to 9.5% error. (5% relative decrease in error)

Gives you an estimate on how much you can improve performance by focusing on this task.

Error Analysis

- A simple manual procedure can save a lot of time and helps in deciding what to focus on.

- Here we evaluated if a single idea is worth focusing on.

Evaluating Multiple Ideas in Parallel

Let's say you have the following ideas:

- Fix pictures of dogs being recognized as cats
- Fix great cats (lions, panthers, etc.) being misrecognized
- Improve performance on blurry images.

Image	Dog	Great Cats	Blurry	Comments
1	X			Pitbull
2			X	
… 100		X	X	Lion on Rainy day at zoo (blurry)
% of total	8%	43%	61%	

Evaluating Multiple Ideas in Parallel

During analysis you might find other reasons for misclassification, which you can add to the spreadsheet

Image	Dog	Great Cats	Blurry	Snapchat/Instagram filters	Comments
1	X			X	Pitbull
2			X		
… 100		X	X	X	Lion on Rainy day at zoo (blurry)
% of total	8%	43%	61%	12%	

Evaluating Multiple Ideas in Parallel

- Gives you an idea of how worthwhile it might be to work on each of these categories of errors.
- In this example blurry and great cats give out a strong vibe of something you want to pursue.
- Doesn’t give you a rigid mathematical view of what to pursue, but gives you a sense of direction in which you might want to move.
- It also helps in deciding which options are not worth pursuing.

Failure Modes in Deep Learning

Adapted from CIS 700: Deep Learning for Data Science, taught by Prof. Konrad Kording, David Rolnick and Jeffery Cheng

Let’s get our hands dirty

Failure Modes in Deep Learning

Symptom: Network won't go above random performance.

- **Cause:** Poor architecture (too narrow or deep), see here and here
  - **Diagnostic:** Increasing width or decreasing depth fixes issue
- **Cause:** Hasn't started learning yet
  - **Diagnostic:** Wait a couple epochs, plot performance and check significance.
- **Cause:** Poor initialization, see here
  - **Diagnostic:** Try other initializations. If using ReLUs, a good rule of thumb is (non-truncated) He normal initialization for the weights.
- **Cause:** Problem simply not solvable
  - **Diagnostic:** Would a human be able to solve it with lots of time? If not, good sign deep learning may fail. Also check completely different architectures and hyperparameters.
- **Cause:** Poor set of input features.
  - **Diagnostic:** Should the data be normalized? Might preprocessing the data in some way make the solution easier to find/express?
- **Cause**: Implementation bug (i.e., in your code).
  - **Diagnostic:** Can you overfit a small subset of the training data?

Failure Modes in Deep Learning

Symptom: Loss bounces over epochs.

- **Cause:** Learning rate too high
    - **Diagnostic:** Try decreasing the learning rate.

- **Cause:** Minibatch size is too small.
    - **Diagnostic:** Essentially equivalent to a large learning rate. Increasing the minibatch size or decreasing the learning rate should fix the problem.

Failure Modes in Deep Learning

Symptom: Catastrophic forgetting - i.e. learning how to solve a task and then forgetting it later in training.

- **Cause:** Data is presented sequentially instead of intermixed.
    - **Diagnostic:** Does shuffling the data fix the problem?

Failure Modes in Deep Learning

Symptom: Suspiciously good performance

- **Cause:** Encoded an answer in the input
    - **Diagnostic:** Look through input carefully and see whether cheating is possible.

- **Cause:** Test dataset too small
    - **Diagnostic:** Is there stochasticity in test accuracy between training runs?

Failure Modes in Deep Learning

Symptom: Plateaus at a strangely low performance.

- **Cause:** Learning rate too low.
    - **Diagnostic:** Try increasing the learning rate.
- **Cause:** Not using a good optimizer
    - **Diagnostic:** Try a different one, e.g. Adam instead of SGD. Can also change the hyperparameters within the optimizer.
- **Cause:** Too much regularization
    - **Diagnostic**: Try less regularization, for example, by decreasing the dropout proportion.