ML strategy ⇒ The systemetic method for improving your model's performance.

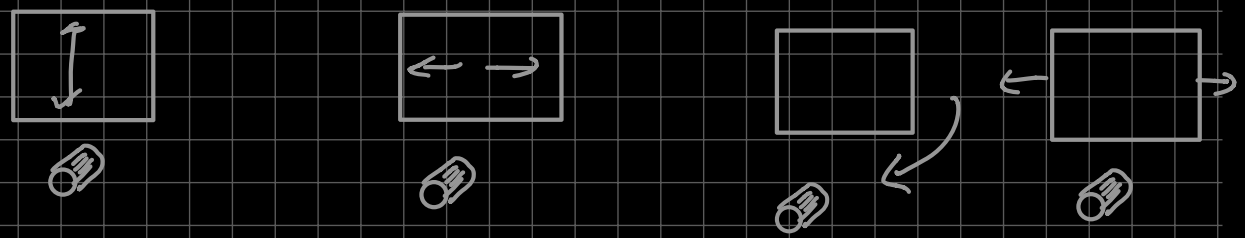
Ways of analyzing a ML problem that will point you in the most promising direction

Ideas:

- collect more data
- collect more diverse training set
- Train algorithm longer with gradient descent
- Try Adam instead of gradient descent / RMSprop
- Try bigger network
- Try drop out
- Add L_2 regularization
- Try different Network architecture
  - Activation functions (relu → selu. elu leaky relu)
  - # hidden units

⇒ One of the biggest challenges in ML is that you have so many things to try, so many hyper-parameters to tune.

I. Orthogonalization ⇒ Those most effective ML people have very clear-eyed about what to tune in order to achieve one effect. (E.g., the TV producer design knots to control different display effect of a TV.



One knot to tune one effect. If a knot can simultaneously tune multi-effects, then it's hard for us to get the picture in the center.
① ②

① ⇒ ML Hyper-parameters

② ⇒ ML goal.

Orthogonalization in ML:

- Fit well on training set
  - bigger network
  - Adam
  - ⋮
  - early stopping
- Fit well on development set
  - Regularization
  - Bigger training set
  - early stopping
- Fit well on test set
  - Bigger dev set
  - class weights
- Fit well in real world
  - change dev set
  - Change cost function

II. Setting up the right goal:

- Single number evaluation metric
- Single optimization metric and many satisfying metrics
- Dev, test set in the same distribution
- Set the right size of dev-test set-
  - big enough for confident evaluation
- Know when to change dev/test sets and metrics.

III. Human level error:

Human level error

Training error

Dev error

- Avoidable bias ①
- Variance ②

① > ②, let model overfit

② > ①, add regularization