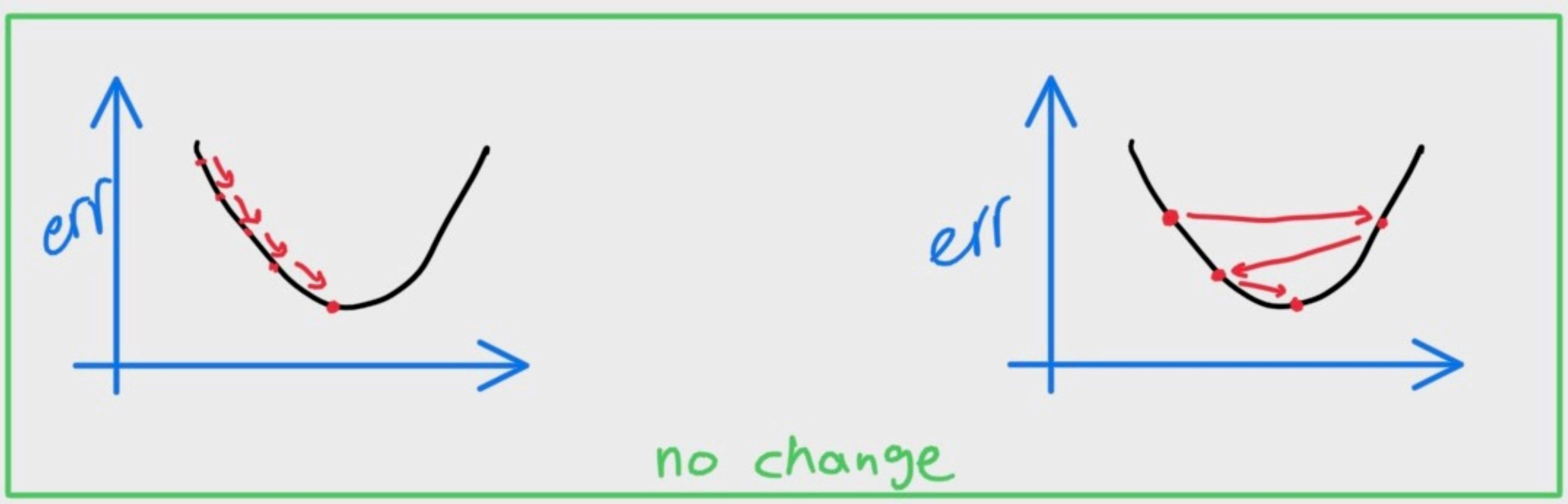
Tuning Deep Learning Models

- optimizer Hyperparameters
  learning rate, batch size, epochs
- Model Hyperparameters
  number of layers, hidden units, model-specific

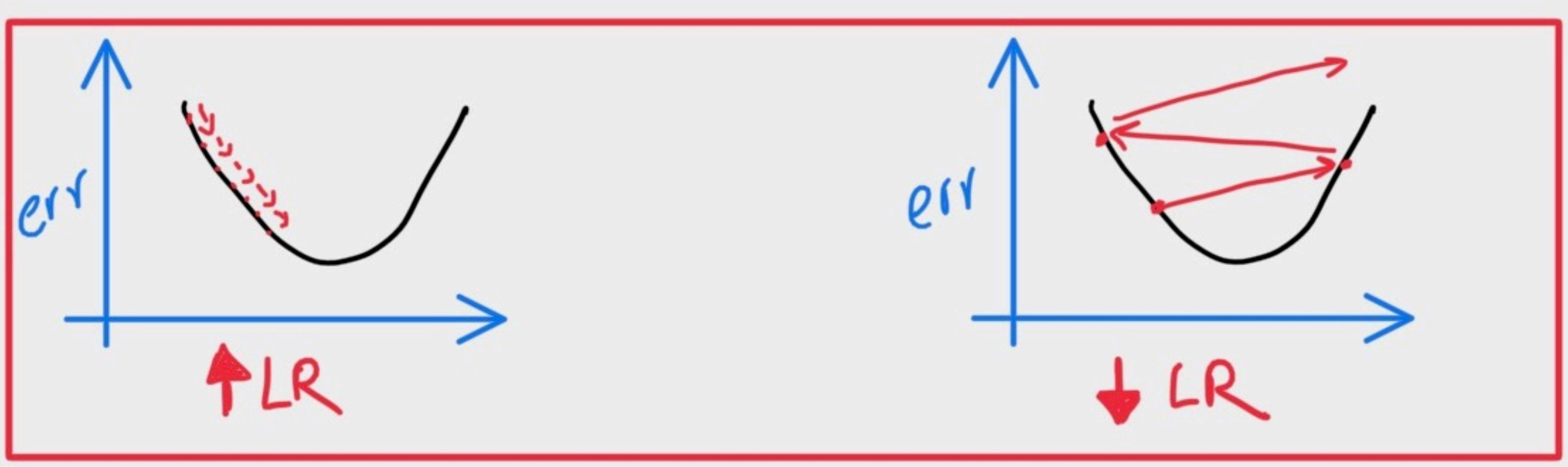
Learning Rate

Seemingly most important hyperparameter.
Good initial value = 0.01



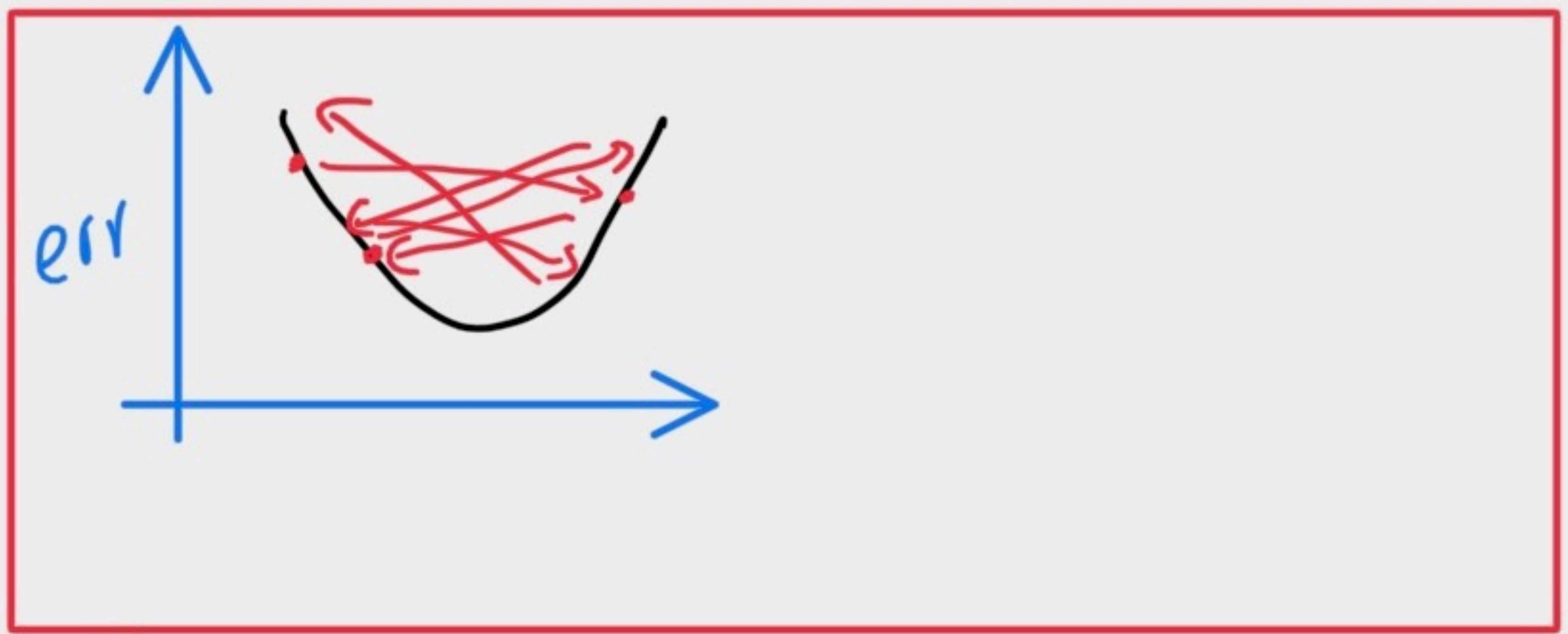
Good Cases

Validation Error decreases



Bad Cases

Validation Error Increases or does not decrease fast enough



Bad Case

Validation Error stops decreasing and somehow osillates.

Solutions → learning rate decay

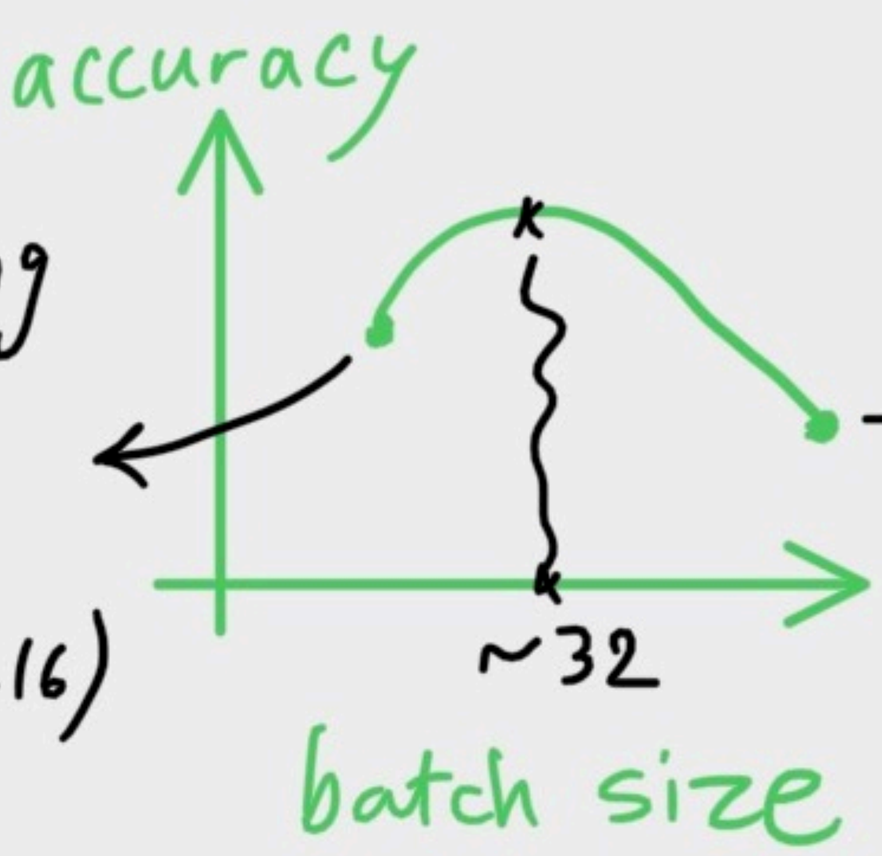
- ↓ LR linearly (divide by half every k epochs)
- ↓ LR exponentially (mult by 0.5 every k epochs)
- Adaptive Learning → ↓/↑ as needed

Batch Size

Affects both resource requirements and training
A good initial value ~ 32, 64

↑ Batch Size requires more memory.

↓ Batch size has more noise which helps the GD not to get stuck in local minima.

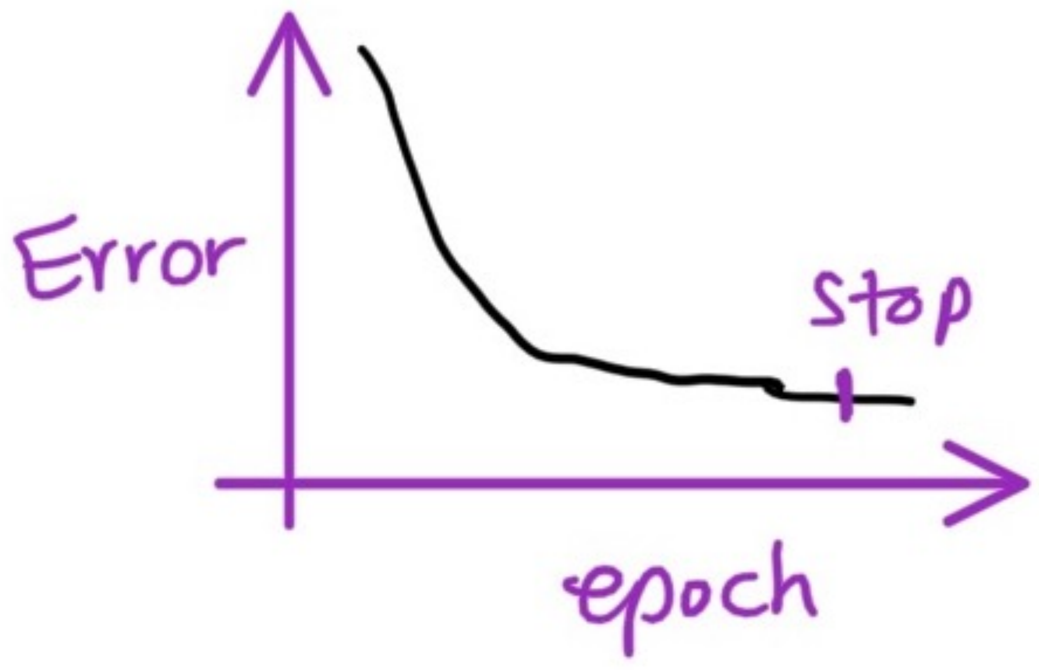


training too slow (1, 2, ..., 16)

Computationally taxing
poorer performance

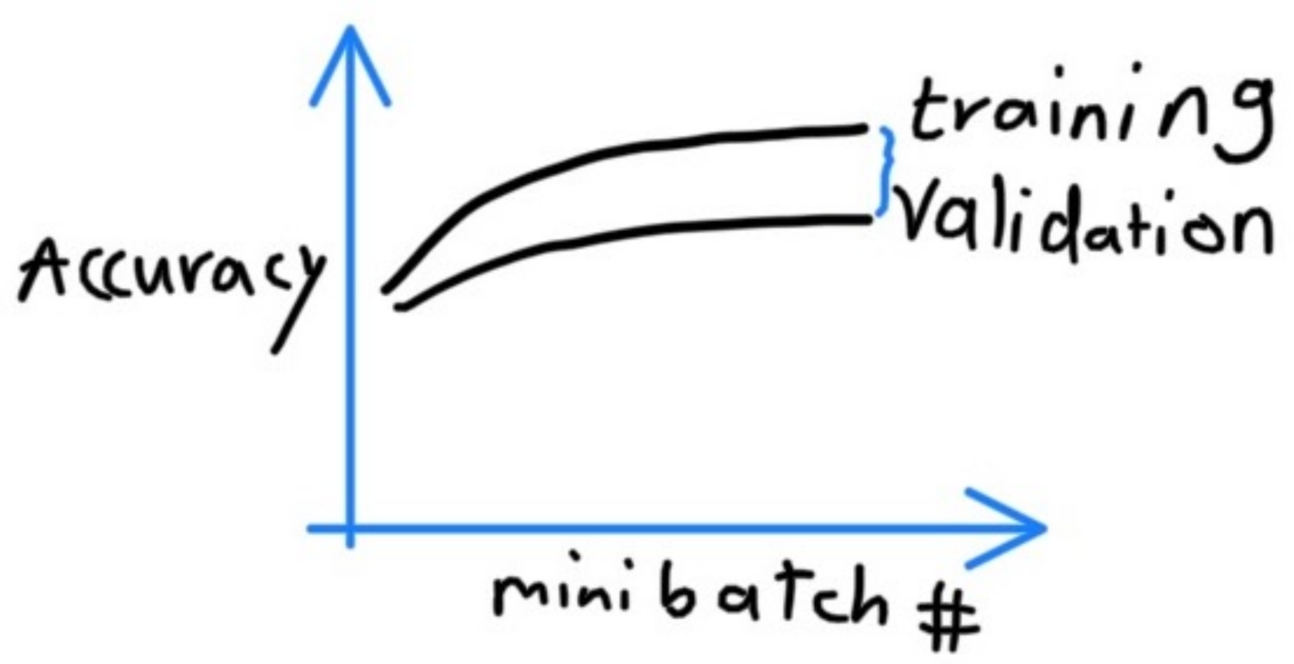
epochs

- we must monitor validation loss when it stops decreasing, we should stop.
- early stopping technique: stop training if the validation loss has not decreased in k epochs.



Model Size

- number of hidden units. Intuitvly, it controls the models capacity to learn a function.
- Having too much capacity → overfit



⇝ overfitting

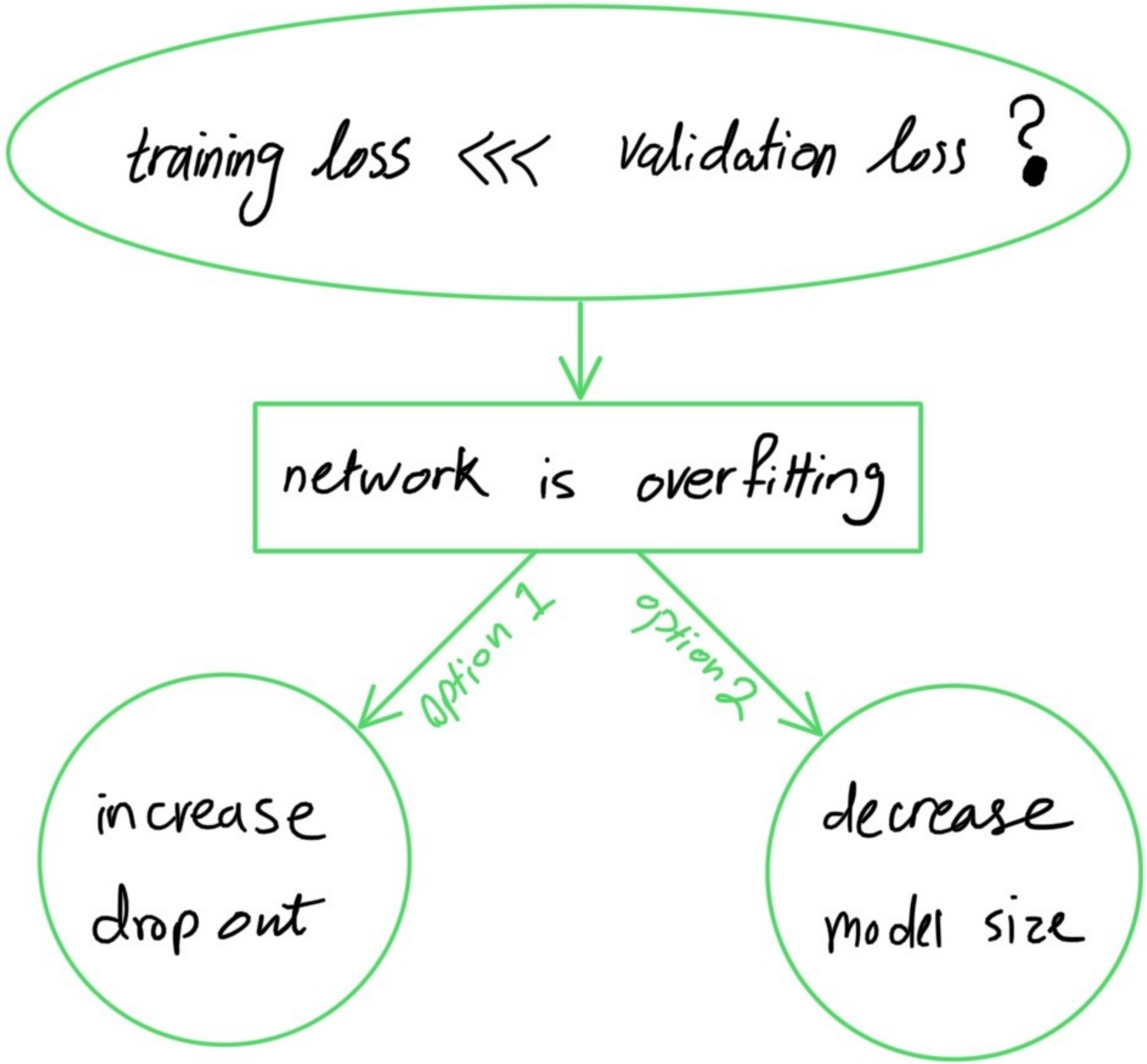
- But the general rule is "More is Better"

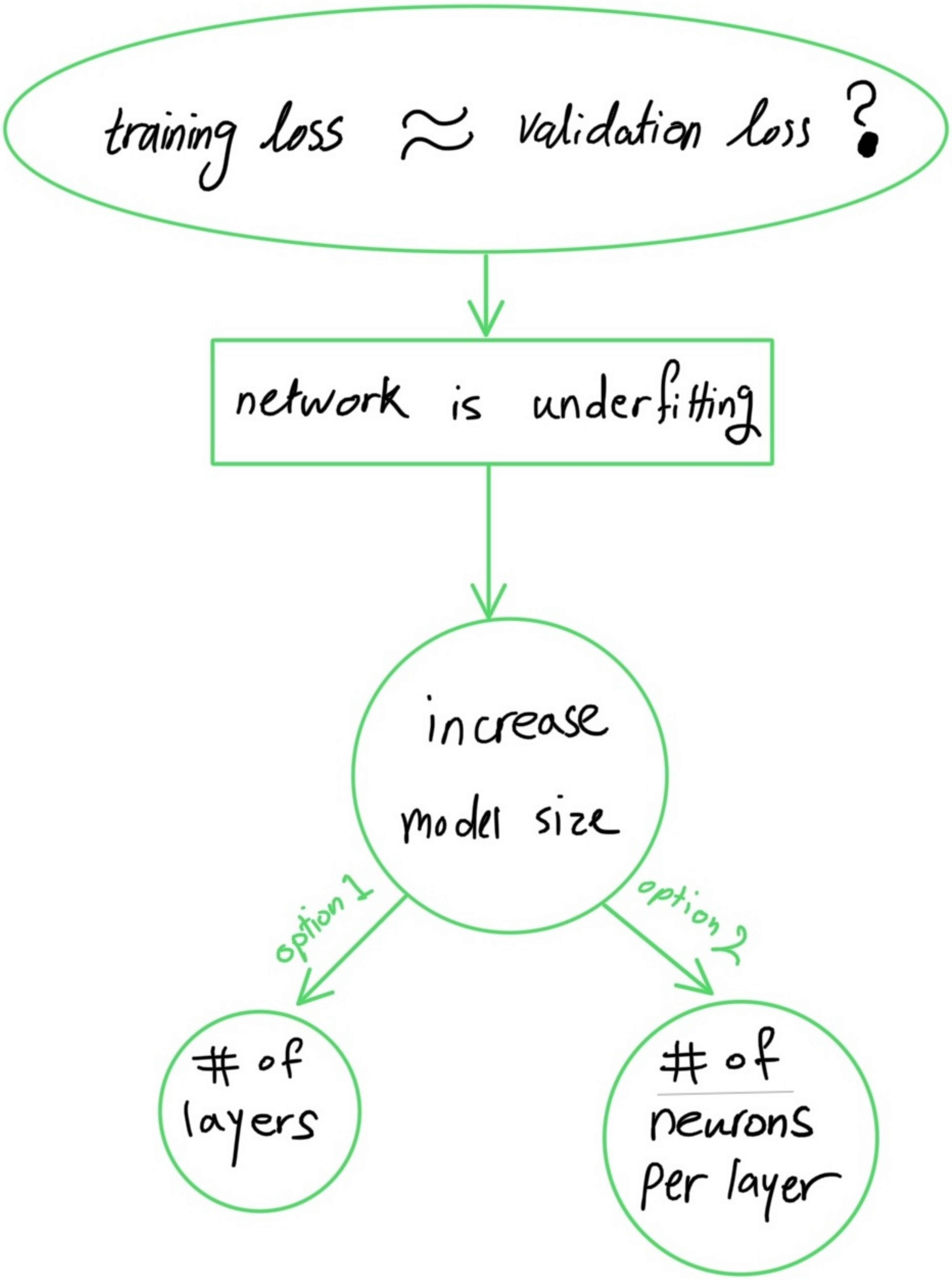
- Number of layers. As a rule of thumb, 3 layers is better than 2. But more than that does not usually help. (exception: CNN)

Cell Type

- In an RNN, we can choose LSTM, GRU, or just the vanilla RNN. The first 2 are usually better.
- Choosing LSTM or GRU is task-dependent.

Selecting Reasonable Hyperparameters





- the number of model parameters should be about same magnitude as the size of dataset.

100 MB dataset (~ 100 million chars)

number of model params $\ll$ size of dataset ?

the model underfits