Hyper-parameter optimization

Presented By: Aaron D. Saxton, PhD
The many layers of practical machine learning

- Model has trainable parameters
  - Weights, kernels, etc
- Parameterize model architecture
  - Number of hidden layers, kernel sizes, etc.
- Training algorithm has parameters
  - Learning rate, learning rate decay, batch size, etc.
- Dataset can have parameters
  - Randomization scheme/seed, train-validate partition, data distortion, adversarial data.

- Anything that isn’t a model’s trainable parameter is a “Hyper-Parameter”
Need a target function: When is a model “Good”?

• Partition dataset into Train, Validate, Test (60, 30, 10% ?)
• A model can be “good” when
  • Accuracy good on validation set
  • Loss good on validation set
  • Training time to solution is good
  • Generalization is good (advanced topic)
• Training loop will have to include periodic validation inference without optimizing model weights
• For argument sake, our target function will be validation loss after a fixed number of training steps (or epochs)
• Testing dataset is reserved to test your final optimal model
  • Emphasis: Testing set should not be used while training or performing hyper parameter optimization
Optimization Problem

- In principle, you could make Hyper-parameters a second set of trainable parameters and a loss function that is validation loss of the last training step and use SGD.
  - Back-prop would include every iteration of weights for every training step over many copies of training data.
  - Simply put: Computationally Prohibitive
  - Must use methods beside gradient methods
Popular Hyper-parameter optimization algorithms

- Manual Search
  - Practitioner tries combinations of hyper-parameters, uses intuition and experience to guide.
- Grid Search
  - sample hyper-parameters at regular intervals
- Random search
  - Use a distribution to randomly choose sets of hyper-parameters
- Genetic Algorithm
Random Search

- Each point in the plot is one “experiment”.
  - Fix hyper-parameters, fixed number of training epochs, measure validation loss
- Grid layout: Samples parameter with redundant value
- Random Layout: Same number of experiments will sample parameter many more times
- Good improvement Still just guessing

Random Search for Hyper-Parameter Optimization, J. Bergstra, Y Bengio, 2012
Genetic Algorithms

• Choose an initial population of hyper-parameter tuples and perform an experiment with each one
  • e.g. (learning rate, batch size, l2 coeff)
    (.0002, 32, .1)
    (.0005, 128, .5)
    (.0001, 64, 1)
• Choose the tuples that give the lowest final validation loss
  • For argument sake, say (.0002, 32, .1), (.0005, 128, .5)
• Create “offspring” by choosing and or combining traits from the “parents”
  • Many different ways of making “offspring”
  • One example (.0002, 128, .1), (.0002, 32, .5), and (.0005, 128, .1) as children of the above parents.
• Mutate! one or many parameters and/or children
  • e.g. (.0003, 128, .1), (.0002, 32, .5), and (.0005, 128, .1)
• Perform experiment on new hyper-parameter population
• Further reading: encode your hyper parameters with “Gray Encoding”