A Few Notes on Improving Generalization

Gary Cottrell

Deep Learning

Recall from Week 1: How to deal with overfitting?

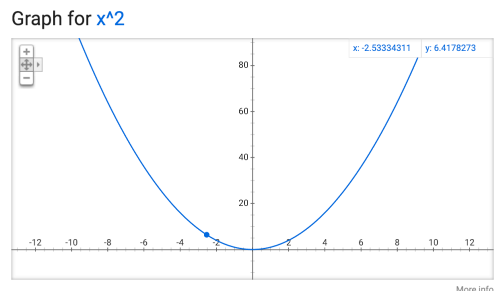
- The best way: *Get more data!* (Or, manufacture more data...) this is what all those transformations in PyTorch do.
- Minimize $J=E+\lambda C$ where E is the error and C is a measure of model complexity (*regularization*).
- Early stopping:
  - Have a hold out set (some fraction of the training set) – this is a stand-in for the unseen test set
  - Use the remaining portion of the training set to change the weights
  - Watch the error on the holdout set and stop when it starts to rise.

What form can C take?

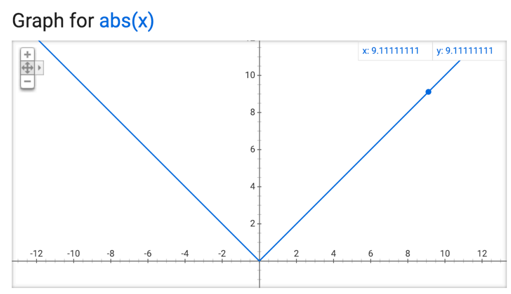
- General Idea: Make the model "smaller"
  - L_2 regularization: Minimize $\|W\|_2^2$
    - Derivative: 2W - Make the weight smaller in proportion to its size,
  - L_1 regularization: Minimize $|W|$
    - Derivative: 1 –Make the weight smaller at a constant rate.
  - Rumelhart's idea: Minimize $C = ||w||^2 / (||w||^2+1)$
    - Penalizes big weights less while penalizing small weights more, driving them to 0.

What form can C take?

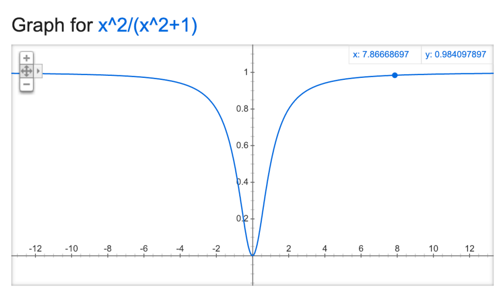
- L_2 regularization:
  - Big weights more



- L_1 regularization:
  - All weights the same



- Rumelhart's idea:
  - Small weights more



Another idea: Dropout

- For any hidden layer, probabilistically *turn off* some fraction of the hidden units

- This makes it so that no hidden unit can depend on any other

- Explores an exponential number of models

Another idea: Dropout

- Explores an exponential number of models
- Assume dropout rate is 0.5
- What happens if we have 4 hidden units:
  - Can have (Assuming *exactly* half turn off)
    - 0011
    - 1001
    - 1010
    - 1100
    - 0101
    - 0110
  - This is 4 choose 2 (6)

Another idea: Dropout

- 4 choose 2: 6
- 8 choose 4: 70
- 10 choose 5: 252
- 12 choose 6: 924
- 100 choose 50: e^{29}
- So if exactly half turn off, we won't actually investigate an exponential number of models, (unless we train for e^{29} epochs...) but we'll try.

Add noise to the training set or the model:

- Add a small amount of gaussian random noise to the inputs
- Add a small amount of gaussian random noise to the hidden unit activations
- Both make the model more robust to perturbations: makes them generalize better.

So: to improve generalization

- Early stopping
- Get more data or create more data artificially
- Complexity minimization (regularization)
- Dropout
- Add noise to the input or the model
  - (I don’t know if anyone has tried adding noise to the output...)