# Deciding What to Try Next *Revisited*

*Evaluating a Learning Algorithm*

Advice for Applying Machine Learning:

**Debugging a learning algorithm:**

Suppose you have implemented regularized linear regression to predict housing prices. However, when you test your hypothesis in a new set of houses, you find that it makes unacceptably large errors in its prediction. What should you try next?

**Debugging a learning algorithm:**

Suppose you have implemented regularized linear regression to predict housing prices. However, when you test your hypothesis in a new set of houses, you find that it makes unacceptably large errors in its prediction. What should you try next?

- Get more training examples
- Try smaller sets of features
- Try getting additional features
- Try adding polynomial features $(x_1^2, x_2^2, x_1x_2, \text{etc})$
- Try decreasing $\lambda$
- Try increasing $\lambda$

# Actions

- Some actions
  - Get more training examples
  - Try smaller set of features (a small set of features)
  - Try getting additional features (just the opposite)
  - Try adding polynomical features
  - Try decreasing lambda
  - Try increasing lambda

# Actions

- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features)
  - Try getting additional features (just the opposite)
  - Try adding polynomical features
  - Try decreasing lambda
  - Try increasing lambda

# Actions

- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features) (fixes high variance)
  - Try getting additional features (just the opposite)
  - Try adding polynomical features
  - Try decreasing lambda
  - Try increasing lambda

# Actions

- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features) (fixes high variance)
  - Try getting additional features (just the opposite) (fixes high bias)
  - Try adding polynomical features
  - Try decreasing lambda
  - Try increasing lambda

# Actions

- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features) (fixes high variance)
  - Try getting additional features (just the opposite) (fixes high bias)
  - Try adding polynomical features (fixes high bias)
  - Try decreasing lambda
  - Try increasing lambda

# Actions

- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features) (fixes high variance)
  - Try getting additional features (just the opposite) (fixes high bias)
  - Try adding polynomical features (fixes high bias)
  - Try decreasing lambda (fixes high bias)
  - Try increasing lambda

# Actions

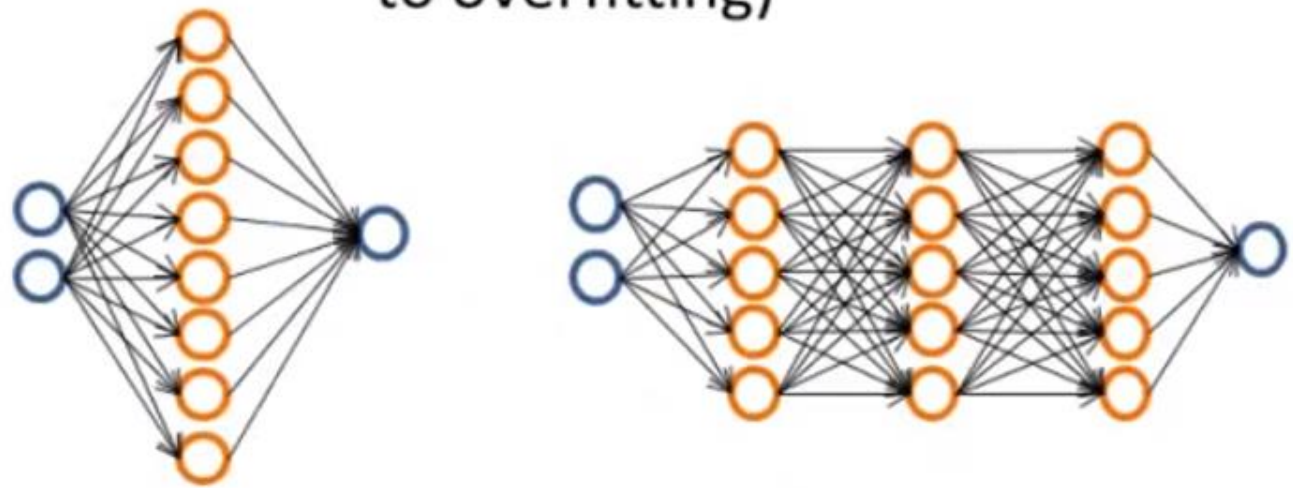
- Some actions
  - Get more training examples (fixes high variance)
  - Try smaller set of features (a small set of features) (fixes high variance)
  - Try getting additional features (just the opposite) (fixes high bias)
  - Try adding polynomical features (fixes high bias)
  - Try decreasing lambda (fixes high bias)
  - Try increasing lambda (fixes high variance)

# Neural networks and overfitting

"Small" neural network
(fewer parameters; more prone to underfitting)

Computationally cheaper

"Large" neural network
(more parameters; more prone to overfitting)



Computationally more expensive.

Use regularization ($\lambda$) to address overfitting.

# Exercise

- Suppose you fit a neural network with one hidden layer to a training set. You find that the cross validation error $J_{CV}(\theta)$ is much larger than the training error $J_{train}(\theta)$. Is increasing the number of hidden units likely to help?
- Yes, because this increases the number of parameters and lets the network represent more complex functions.
- Yes, because it is currently suffering from high bias.
- No, because it is currently suffering from high bias, so adding hidden units is unlikely to help.
- No, because it is currently suffering from high variance, so adding hidden units is unlikely to help.