Neural Networks and Deep Learning

Regularization

Nicholas Dronen

Department of Computer Science
dronen@colorado.edu

February 6, 2019

University of Colorado **Boulder**

Bias-Variance Tradeoff

Regularization Techniques
- Data Augmentation
- Early Stopping
- Weight Penalties
- Dropout
- Miscellanous

ICLR 2017 Best Paper

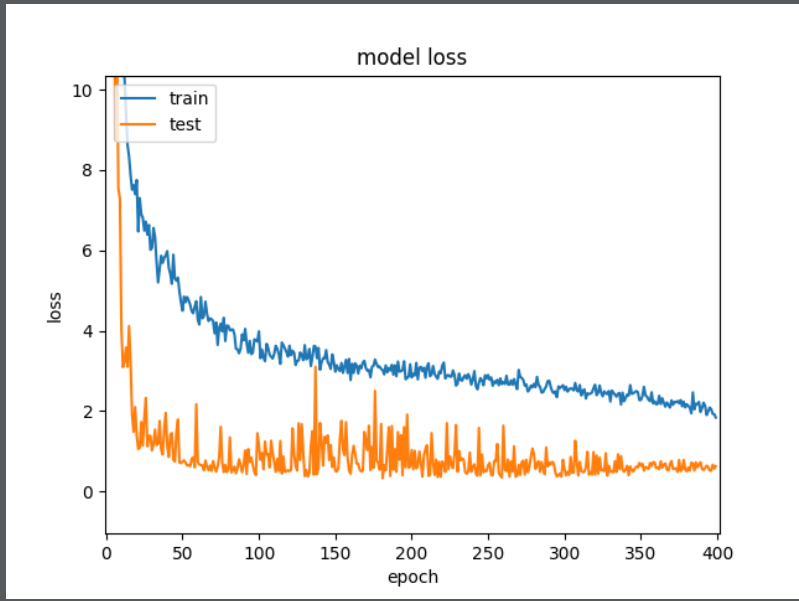
Understanding Deep Learning Requires Rethinking Generalization, Zhang, et al, ICLR 2017

Our central finding can be summarized as:
Deep neural networks easily fit random labels

Finding the Best Hypothesis

Characteristics of a good model

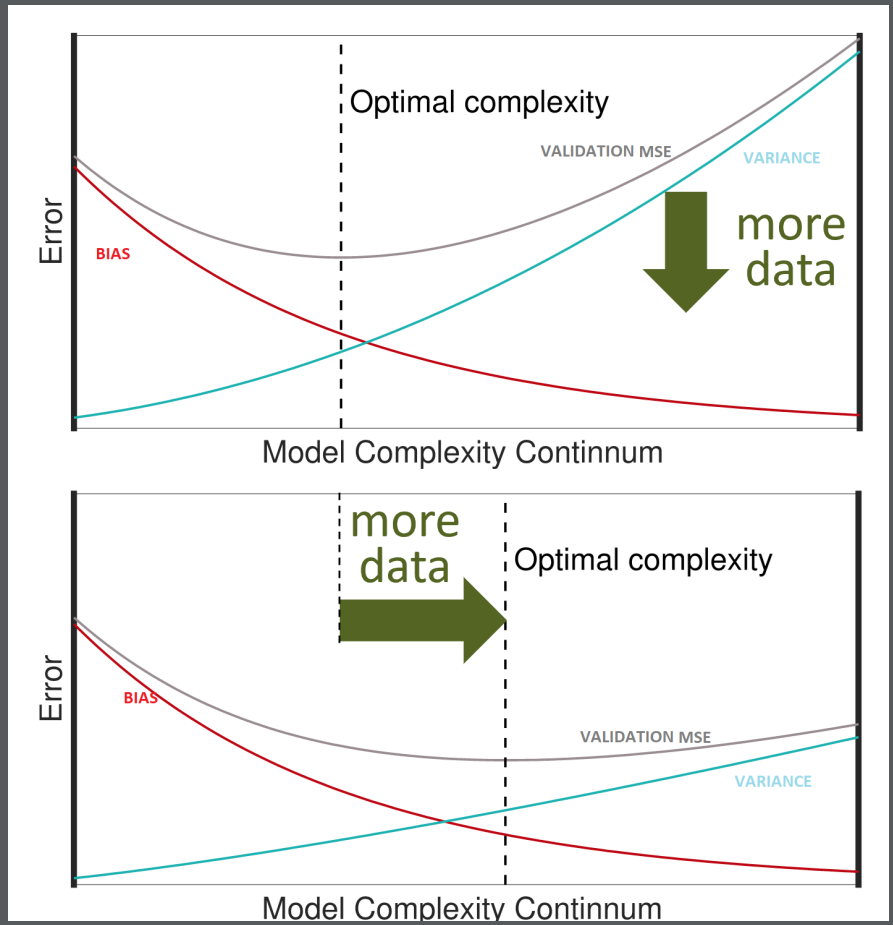
- Minimum error on validation set
- Good approximation on test set



Approximating Complex Functions

- Increase model complexity
- Massive amounts of training data
- If both the criteria are satisfied, we are finished.
- This rarely happens outside labs.

Approximating Complex Functions

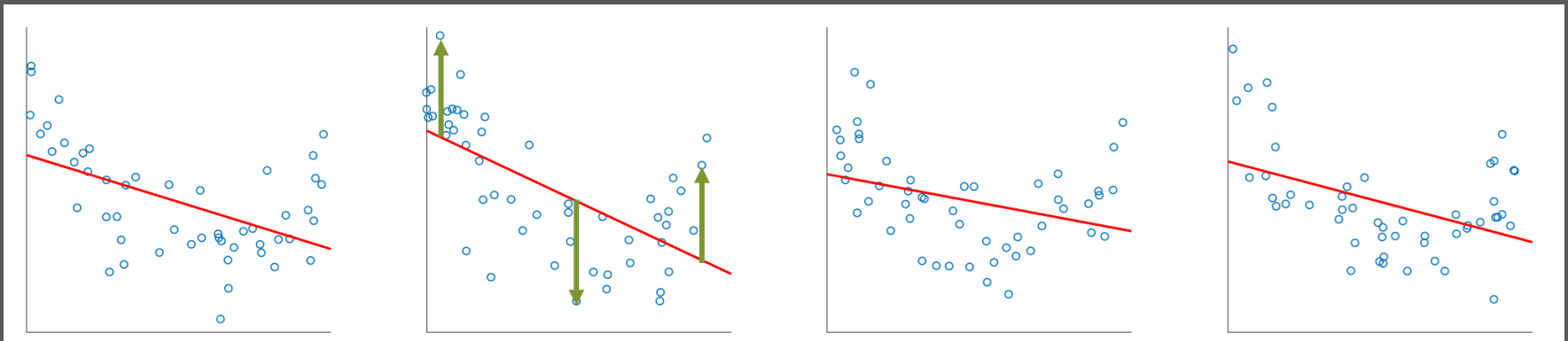


Bias and Variance

- Approximating complex functions using complex hypotheses is risky.
- Leads to overfitting or underfitting if above 2 criteria are not satisfied.
- Bias and variance can be used as a tool to identity underfitting and overfitting

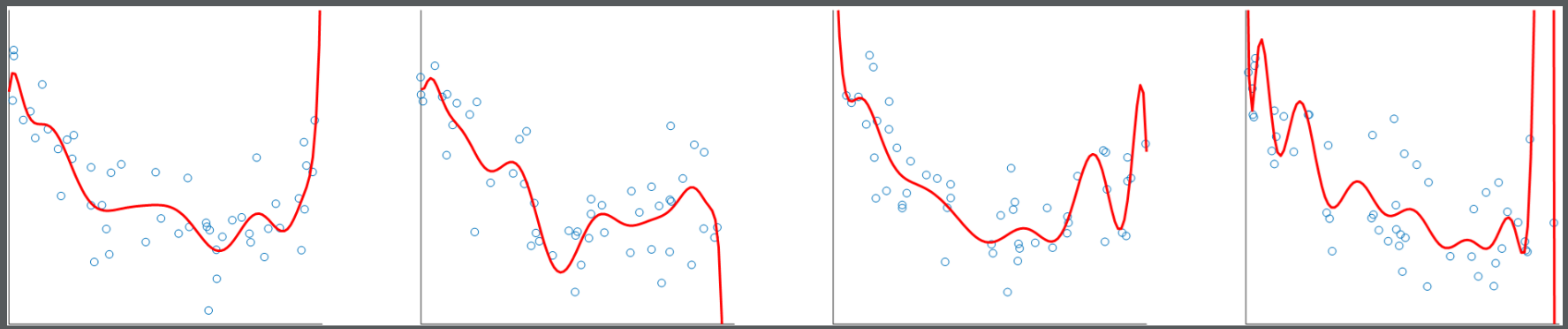
Bias

- Regardless of training sample size or characteristics, the model will produce consistent errors.
- Solution: Increase model complexity. Make it more deep/expressive.



Variance

- Different samples of training data yield different model fits.
- Resulting model trained on 1 set needs to be retrained on other set.
- Solution: Increase data size or simplify model.



Bias-Variance Decomposition

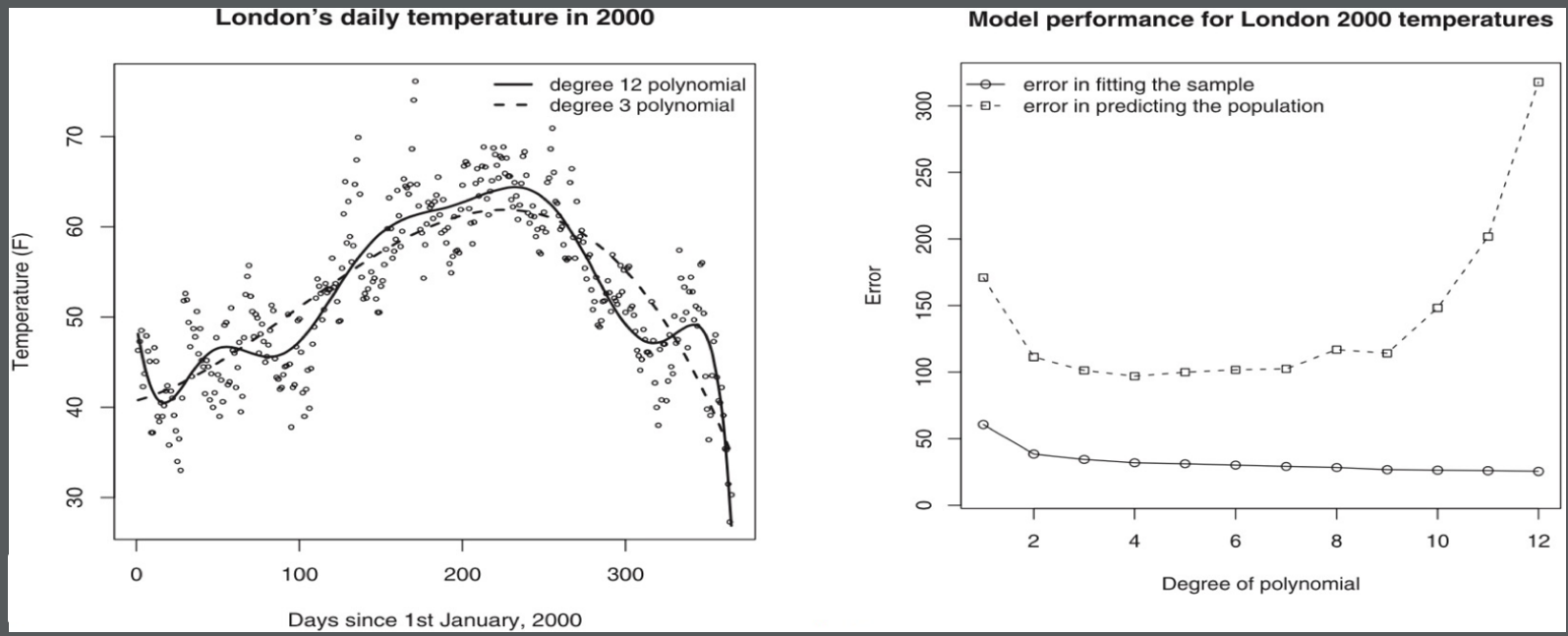
- Given a data set $D = (x_1, y_1), ..., (x_n, y_n)$
- A model built from data set D: $f(x; D)$
- We can evaluate model effectiveness using Mean Squared Error:

$$
\begin{aligned}
MSE &= E_{p(x,y,D)}[(y - f(x; D))^2] \\
&= \underbrace{(E_D[f(x; D)] - E[y|x])^2}_{1} + \underbrace{E_D[(f(x; D) - E_D[f(x; D)])^2]}_{2} + \\
&\underbrace{E[(y - E[y|x])^2]}_{3}
\end{aligned}
$$

1. bias - difference between average model prediction (across data sets) and the target
2. variance of models (across) data sets for a given point.
3. intrinsic noise in data set

Bias Variance Dilemma - Model Selection

- Use bias/variance analysis as a tool for model selection.
  - » If bias is high: Add more layers to the network
  - » If variance is high: Decrease the layers in the network
- Can be approximated using many variants of D i.e. Bootstrap Sampling



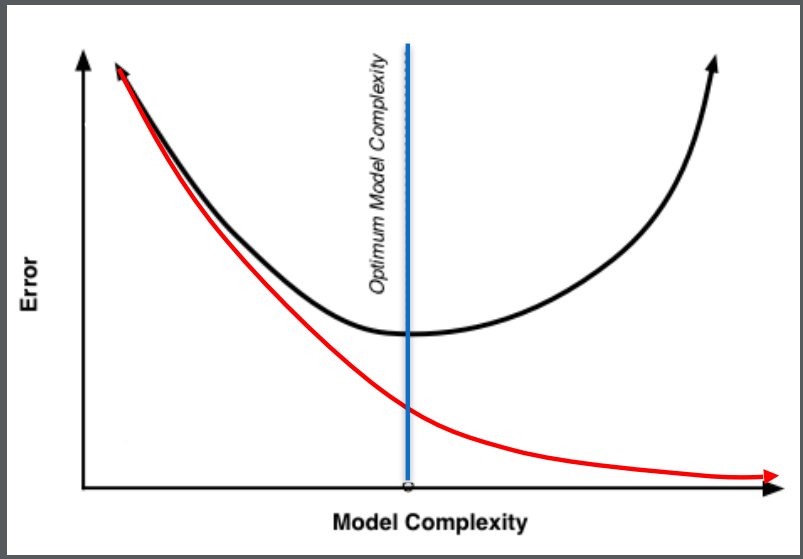
Bias Variance Analysis - Conclusion

- High Bias
  - Choose a more complex model
  - Train model for higher number of epochs
- High Variance
  - Bootstrap Aggregation
  - Batch Normalization
  - Data Augmentation
  - Regularization

Setting Model Hyperparameters

How do you select the optimal number of hidden units, number of layers, connectivity, etc.?

- Validation method: Split training set into two parts, T and V, train many different architectures on T, and choose the architecture which minimizes error on V.
- K-fold cross validation



How do you search over hyperparameters efficiently?

- Early stopping
- Bayesian optimization

The Danger of Minimizing Network Size

Local Optima arise only if you use a highly constrained network

- minimum number of hidden units
- minimum number of layers
- minimum number of connections

Having sparse capacity in the net means there are many identical solutions

- Example: If you have 10 hidden and need only 2, there are 45 equivalent solutions

Basic Strategy

Instead of starting with the smallest net possible, use a larger network and apply various tricks to avoid overfitting.

8 ideas to follow....

Data Augmentation - Visual Data

Augmenting an existing dataset can, in effect, create more examples → better generalization.

- Mirroring
- Random cropping
- Rotation
- Color shifting

Data Augmentation - Text Data

Augmenting an existing dataset can, in effect, create more examples → better generalization.

- Replacing words with synonyms
- Word dropout (e.g. Deep Unordered Composition Rivals Syntactic Methods for Text Classification)

Early Stopping

- Rather than training until network error converges, stop training early
- Rumelhart
  - » hidden units all go after the same source of error initially → redundancy
- Hinton
  - » Weights start small and grow over training
  - » When weights are small, model is mostly operating in linear regime
- Dangerous: very dependent on training algorithm
  - » Example: What would happen with random weight search?
- While probably not the best technique for controlling model complexity, it does suggest that you shouldn't obsess over finding a minimum error solution.

Weight Penalties

L2 weight decay

$$E = \frac{1}{2}\sum_j (t_j - y_j)^2 + \frac{\lambda}{2}\sum_{i,j} w_{ji}^2$$

$$\Delta w_{ji} = \varepsilon \delta_j x_i - \varepsilon \lambda w_{j,i}$$

L1 weight decay

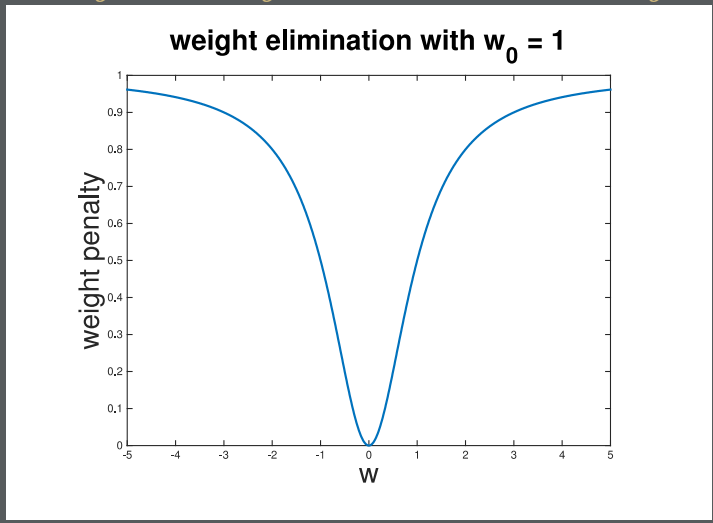
$$E = \frac{1}{2}\sum_j (t_j - y_j)^2 + \lambda \sum_{i,j} |w_{ji}|$$

$$\Delta w_{ji} = \varepsilon \delta_j x_i - \varepsilon \lambda \; sign(w_{ji})$$

Weight elimination

$$E = \frac{1}{2}\sum_j (t_j - y_j)^2 + \frac{\lambda}{2}\sum_{i,j} \frac{w_{ji}^2/w_0^2}{1 + w_{ji}^2/w_0^2}$$

Can be interpreted as Bayesian Priors

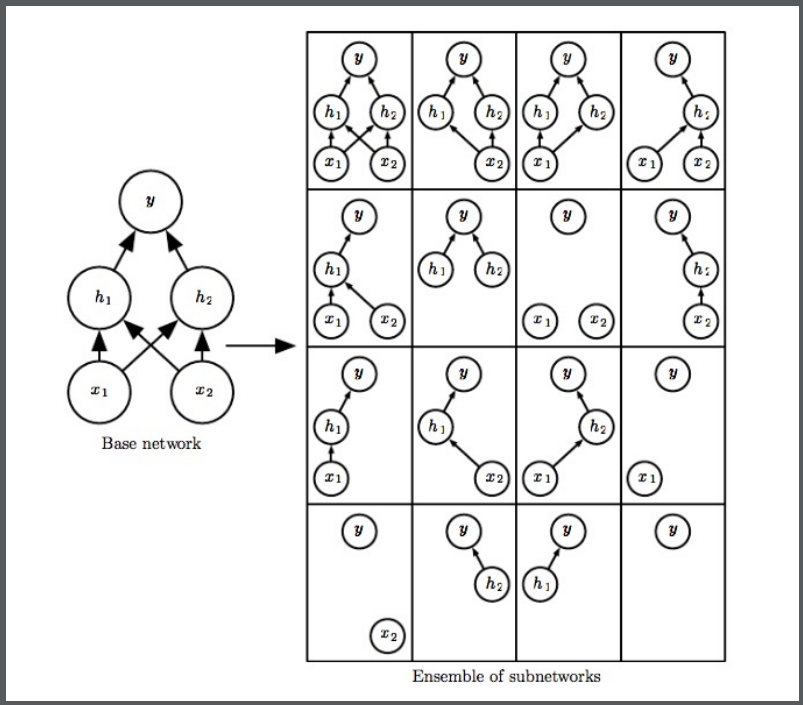


Dropout

1. Dropout
   - » On each training trial, randomly remove a portion of units (hidden and possibly input)
     - leads to robustness of network to lesions
     - leads to less specificity of hidden unit responses, or more sharing of responsibility
     - leads to better generalization
   - » On each test trial, leave all units in, but reweight connections with factor of α (expectation of contribution)

Dropout

Dropout as an ensemble method



Goodfellow, Bengio, Courville 2016

Dropout

With H hidden units, each of which can be dropped, we have 2^H possible models

Each of the 2^{H-1} models which include hidden unit h must share the same weights for the units

- serves as a form of regularization
- makes the models cooperate

Including all hidden units at test with α-scaling is equivalent to computing the geometric mean of all 2^H models

- Exact equivalence with one hidden layer
- "Pretty good approximation" according to Geoff with multiple hidden layers

Dropout

Advantages

- Computationally cheap
- Seems to work robustly, probably better than weight penalties

Disadvantages

- Adds noise to gradient descent, which makes it hard to control learning rates and know when training has reached asymptote

Other Regularization Techniques

- Hard constraint on weights
  - Ensure that $\sum_i w_{ji}^2 < \phi$ for every unit
  - If constraint is violated, rescale all weights $w_{ji} \leftarrow w_{ji} \frac{\phi}{\sum_i w_{ji}^2}$
  - I'm not clear why L_2 normalization and not L_1
- Additive Noise
  - Training with Noise is Equivalent to Tikhonov Regularization

Other Regularization Techniques

- Sparsity Constraints
  - » Penalties on activations, not weights
- Model Averaging
  - » Multiple neural networks
  - » Bagging, boosting, other ensemble methods