MACHINE INTELLIGENCE COMMUNITY

Regularization Notes

October 10th, 2017
By Justin Chen

1. Goal of Learning Algorithms

1.1 What is Generalization

- **Brief recap:**
  - **Neural Networks (NN) are sets of linear equations composed into nonlinear functions**
  - **NN are good for approximating functions that can model nonlinear patterns in data and are good at handling high-dimensional data**
- **The goal of learning algorithm is generalization - learning a set of parameters of a function that can generalize to unseen data (new data not in training set)**
  - **There exists exponentially many sets of parameters that approximate the same function**

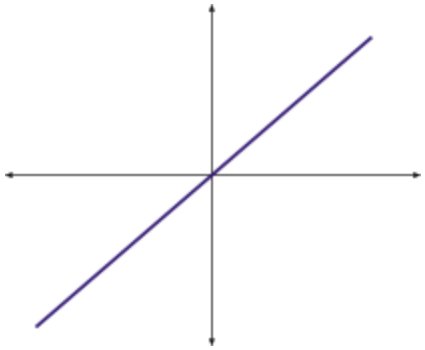
1.2 Factors that influence Generalization

- Training time - number of epochs
- Dataset
  - Sufficient amount of training data
  - Correct labeling
  - Balanced dataset - equal representation of each type of object
  - Features are independent
- Priors
  - Model architecture
    - Certain architectures are better suited for computing on different types of data.
  - Local consistency prior/ Smoothness prior
    - Page 153

2. How Much Data Is Enough Data?

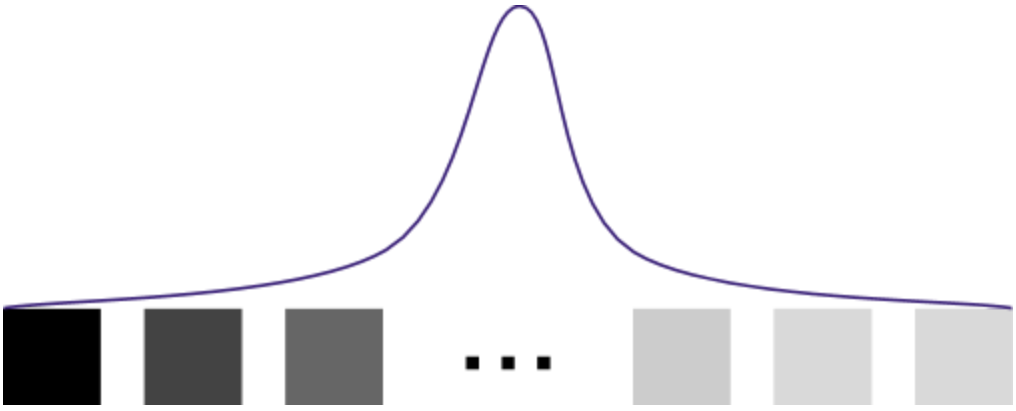
Imagine all your images are single channel 1x1 pixels. Each pixel can take on values 0-255. The total number of possible images would be 256. Let's further assume that in actuality, for this example, the underlying distribution of the pixels is uniform - each pixel have 1/256 chance of occurring.

To model the space of all possible images in this input space would require having 256 sample images. All of these images could be easily modeled by a simple linear equation over a compact set without using nonlinearities.

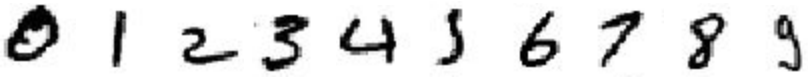


$$f(x) = x$$

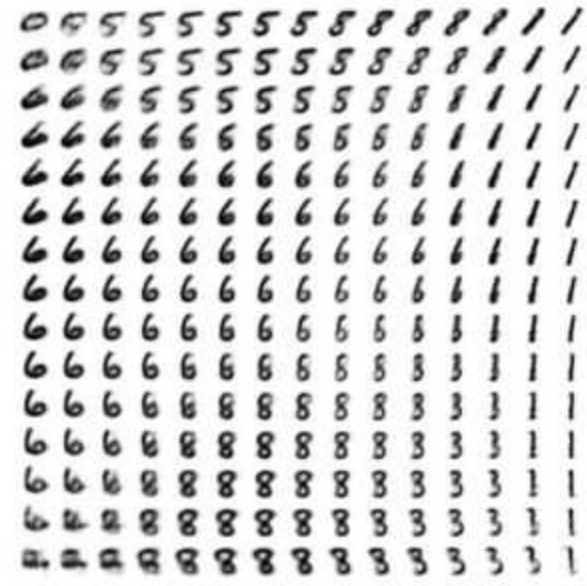
Now imagine the underlying distribution of the pixels is normally distributed centered around 127. You can see that some pixels are less likely to occur that others. Here, pixels with values closer to 0 and values closer to 255 are less likely to occur.



Now let's consider larger images - say images that are 28x28 with a single color channel. The input space here contains 28x28x1x256 = 200,704 possible images. In this space of 28x28 images with one channel, only some of the images are of interest - have any meaning, contain meaningful patterns - the distribution we're interested in. Perhaps in this space, we're interested in images that contain the digits 0-9 such as the MNIST dataset [4].



In this case, we're only interested in the subset of patterns in the total space of 200,704 possible images. In this region, all points form a connected surface. Imagine changing a single pixel of the image of the digit 8. The image will still look like the number 8. As we gradually change more and more pixels, the image will appear less and less like the number 8. You can imagine that as we gradually change these pixels, we are traveling away from the region containing configurations of pixels that form images of the number 8. We refer to this type of region in the input space as a manifold - a connected region in space such that any subset of that region appears to be Euclidean space.



https://i.ytimg.com/vi/nz3otAkzI2Y/hqdefault.jpg

Sampling every point in this total space is expensive and as the dimensionality of our images increases, this becomes ever more intractable and the space becomes sparser. This sparsity makes it harder for learning algorithms to learn the input space (distinguish unique and meaningful regions of the input space). This is also referred to as the curse of dimensionality. Thus we should only sample a representative subset of points from that particular region of the space that we're interested in that captures the true underlying distribution/ pattern. MNIST contains 70,000 images in total and is separated as 60,000 for training and 10,000 for testing (holdout set). Notice that the MNIST dataset represents about 70,000/200,704 = 34.88% of the total space of possible 28x28 images with one channel. 60,000/200,704= 29.89% of that space is used for training. As a reminder, these percentages reflect how much of the total space that was sampled that MNIST occupies , not the percentage of the true distribution - all possible images - of realistic 28x28 handwritten digits. However, the aforementioned metrics serve to give the reader intuition of how data occupies spaces and how much data may be required as dimensionality of the data changes and will serve as a starting point for describing generalization in learning algorithms and why we need to regularize and what it means to regularize. As a sidenote and interesting fact, modern deep

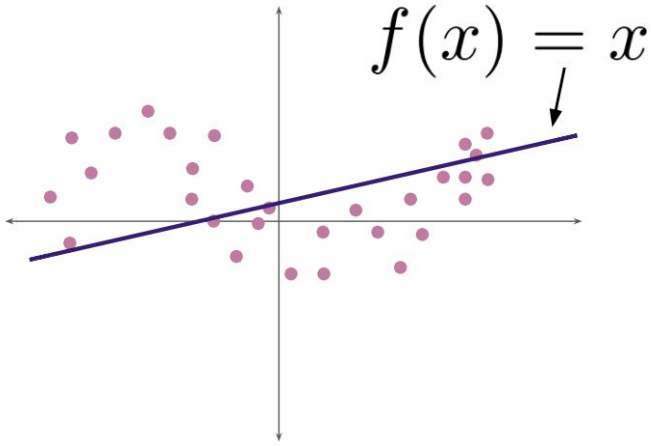
learning algorithms can converge to over 99% generalization accuracy on MNIST in about 6 epochs, which only requires about 2 minutes of training on a modern GPU.

3. Bias-Variance Tradeoff

Machine learning algorithms are data-driven - the quality of learning depends on the quality of data. The quality of a dataset can be measured by its bias, the clustering of the data, and variance, the spread of the data, which can be used to inform how well the algorithm will generalize.

3.1 Bias

- Concentration of data points in a particular region.
- Bias also describes a learning algorithm's prediction error.
- High **bias error** causes underfitting
    - Bias error says "when you predict things, how far are you from the expected **true values**"
    - **Underfitting**
        - **If learning algorithm is suffering from high bias, more training data will not help much. Need to increase number of parameters - model complexity.**
        - https://www.youtube.com/watch?v=e3edL-_fUTo



Data from $f(x) = sin(x - \pi)$

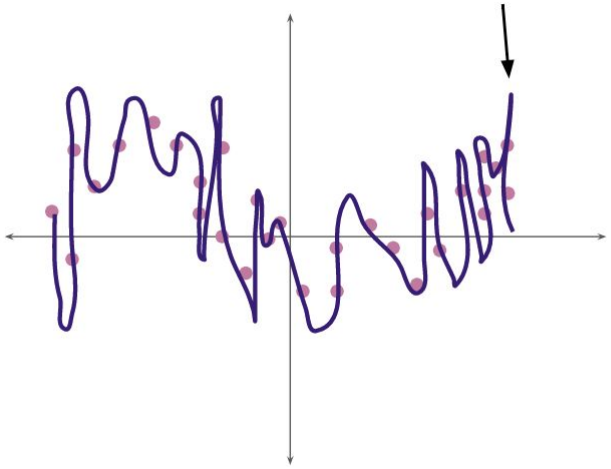
- **Mathematically, bias error measures the difference between the trained model's hypothesis given an input and the expected value for that input.**

$$B = E[f(x; D) - E[y|x]]$$

3.2 Variance

- Spread of data - how far each point is from every other point
- Variance also describes a learning algorithm's prediction error.
- High **variance error** causes overfitting
    - Variance error says "when you predict things how far are you from **other predictions** you've made"
    - **Overfitting**
        - **If learning algorithm is suffering from high variance, algorithm needs more training data and could less parameters (small hidden layers or less layers).**
            - **This type of error is harder to address that underfitting and seen more in practice.**
            - **Could reduce number of features (dimension of input vectors - input space) if model is weighing particular features more than others**

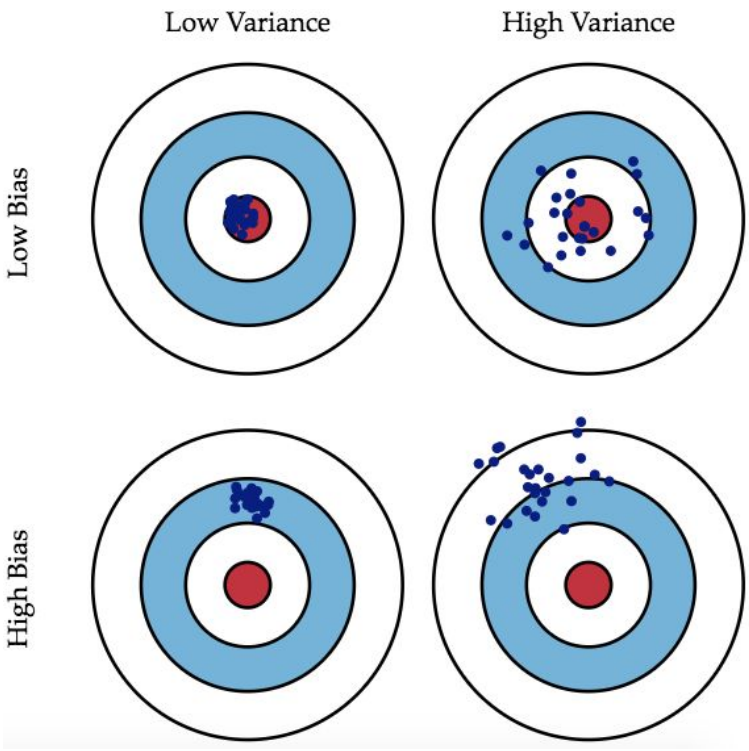
$$f(x) = \theta_n x_n^k + ... + \theta_0 x_0$$



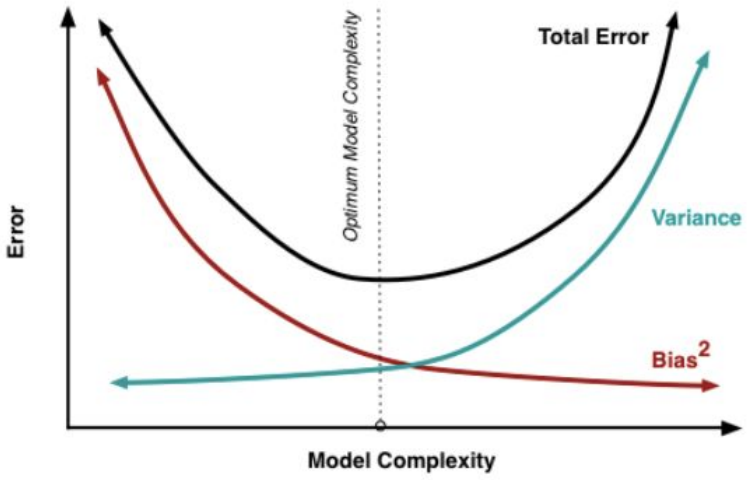
Data from $f(x) = sin(x - \pi)$

- **Mathematically, variance error measures the expected value of the squared difference between the trained model's hypothesis given an input and the expected value for that input.**

$$V = E[(f(x; D) - E[f(x; D)])^2]$$



Bias and Variance in data
http://scott.fortmann-roe.com/docs/BiasVariance.html



Bias and Variance error as a function of model complexity
http://scott.fortmann-roe.com/docs/BiasVariance.html

3.4 Capacity

Capacity describes the model's ability to model data/ learn. More capacity allows a learning algorithm to model more complex functions. There are three different ways to describe capacity.

- **Model capacity - number of trainable model parameters**
- **Representational capacity - set of hypotheses that can be expressed by a trained model's parameters**
- **Effective capacity - set of hypotheses that can be reached by training on a particular set of data**

$$EC(\mathcal{A}) = \{h | \exists \mathcal{D} \ s.t. \ h \in \mathcal{A}(\mathcal{D})\}$$

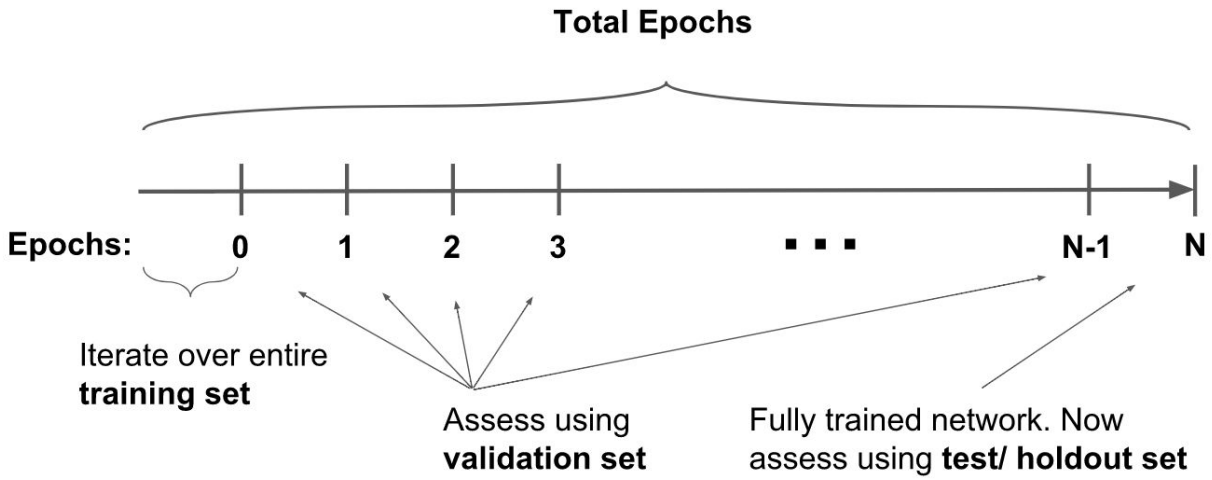
4. Priors

A priori knowledge about the data can aid model selection and training. Input features should be independent so that the algorithm does not learn to over represent features. Additionally, if you know the type of the data, you can select models that are better suited. For example, sequential data like time series, text data, or audio waves are better modeled by architectures like recurrent neural networks that by vanilla feedforward networks. Thus, the No Free Lunch Theorem says that there is no one best learning algorithm that can solve all problems.

5. Regularization

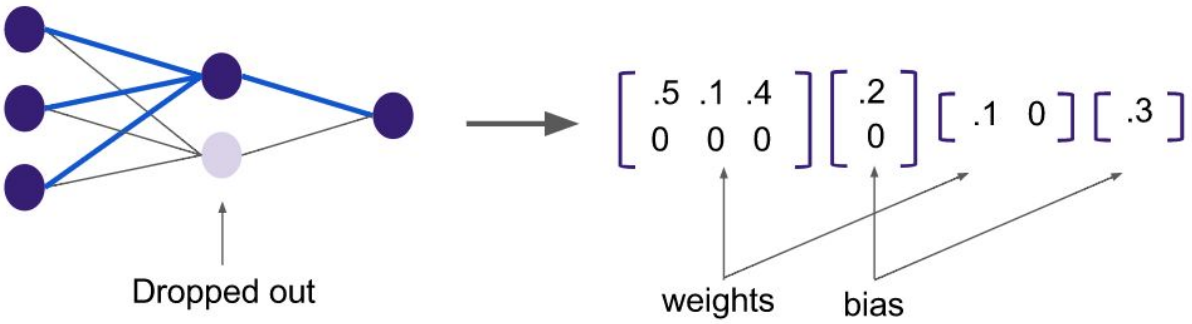
5.1 Early Stopping

- **As training progresses, learning algorithms over train - try to tightly fit their parameters to the training data, which ends up in overfitting.**
- **Training is broken up into epochs, which is one pass over the entire dataset**
- **At the end of each epoch, the model is validated on the validation set to measure accuracy and the model parameters can be saved**
- **Overtime, the validation error should decrease**
- **But if trained for too long, the validation error will increase**
- **Validation error can be tracked overtime and training can stop when the validation error begins to increase**



5.2 Dropout

- **Dropout is the most popular regularization technique**
- **Neurons tend to rely on other neurons to learn, however, we would like the model to learn useful features.**
- **During training on each forward pass, dropout *p*-percent of the neurons in the network - multiply by zero.**
- **Errors are backpropagated normally through the network except neurons that were dropped out receive no gradients**
- **Refer to the paper summary for more details on dropout**



5.3 Weight Decay

As mentioned in the previous workshops, the cost function measures the distance between the model's hypothesis and the correct answer. The cost function directs the learning algorithm towards the correct answer by letting it know how poor its predictions are by penalizing it with a cost - the derivative of the cost function w.r.t. the weights. Models can be regularized by increasing the cost for poor predictions, which is done by simply adding a penalty term to the cost function.

5.3.1 L1 Weight Decay

- **L1-norm is a metric for vectors that measures the absolute distance of each parameter from the origin - from zero**
- **Intuitively, L1-norm measures the total contribution of each feature and regularizing using L1-norm by adding it to the cost tells the model to not overweigh those features.**
- **L1 regularization is usually not applied to bias vectors**
- **L1 regularization tends to sparsify parameter matrices**

$$L1_{Norm} = \|\theta\|_1 = \sum_i |\theta_i|$$

$$\theta := \theta - \frac{\partial}{\partial \theta}(J(x, y; \theta) + \|\theta\|_1)$$

5.3.2 L2 Weight Decay

- **Applies to model weights, but not to bias parameters**
- **L2-norm is the same a Euclidean distance**

$$L2_{Norm} = \|\theta\|_2 = \sqrt{\sum_i \theta_i^2}$$

- **L2 regularization is the L2-norm squared and multiplied by a regularization term, lambda and constant 0.5. The square and 0.5 are for mathematical convenience when taking the derivative.**

$$L2_{Reg} = \|\theta\|_2^2 = \sum_i \theta_i^2$$

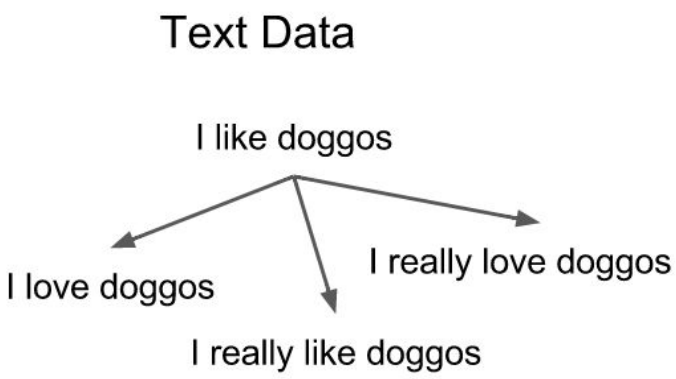
$$\theta := \theta - \frac{\partial}{\partial \theta}(J(x, y; \theta) + \frac{\lambda}{2}\|\theta\|_2^2)$$

$$\theta := \theta - \frac{\partial}{\partial \theta} J(x, y; \theta) - \lambda\theta$$

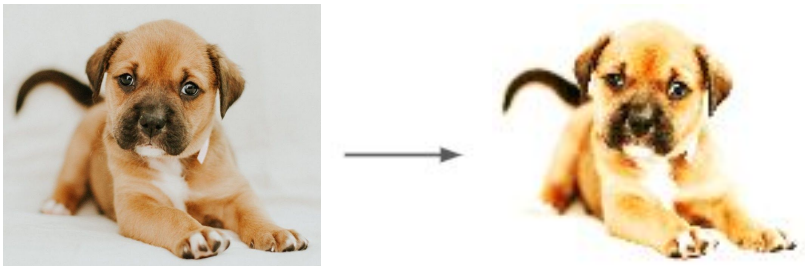
- **Intuitively, L2 regularization tries to balance all features simultaneously and pull them close to the origin diffusing the contribution across all features. This can be thought as trying to utilize the information in each feature to try to gain better generalization error.**

5.4 Data Augmentation

- **Transform existing data to introduce more realistic points to learn on**
- **Increases size of dataset**
- **Transformations are domain-dependent**
    - **e.g. Augment text data by replacing synonyms, adding adjectives, or removing adjectives**



- **e.g. Change the contrast and brightness in natural images**



5.5 Advanced Techniques

- meProp, Swapout, Zoneout, Drop Connect, Stochastic Depth, Adversarial Training

6. Reference & Further Reading

1. https://www.youtube.com/watch?v=e3edL-_fUTo
2. http://scott.fortmann-roe.com/docs/BiasVariance.html
3. http://scott.fortmann-roe.com/docs/MeasuringError.html
4. http://yann.lecun.com/exdb/mnist/
5. **Zhang, Chiyuan, et al. "Understanding deep learning requires rethinking generalization." arXiv preprint arXiv:1611.03530 (2016).**
6. **Krueger, David, et al. "Deep Nets Don't Learn via Memorization." (2017).**
7. **MLA Arpit, Devansh, et al. "A closer look at memorization in deep networks." arXiv preprint arXiv:1706.05394 (2017).**
8. **Halevy, Alon, Peter Norvig, and Fernando Pereira. "The unreasonable effectiveness of data." IEEE Intelligent Systems 24.2 (2009): 8-12.**