Regularization and Overfitting

Shrishti Saha Shetu

University of Erlangen-Nürnberg
Institute for Digital Communications

February 01, 2019

Outline

Background

- A common issue in machine learning is overfitting.
- Occurs when model also captures the noise in a dataset.
- Regularization helps reducing overfitting within our model.

Challenge in Machine Learning

Challenge:

The central challenge in machine learning is to achieve better generalization.

Generalization Error:

- The generalization error is defined as the expected value of the error on a new input.
- It is measured on a test set of examples that were collected separately from the training set.

Capacity

Definition and Effects

Model's capacity is its ability to fit a wide variety of functions.

- Models with low capacity may struggle to fit the training set
- Models with high capacity can overfit by memorizing properties of the training set.

There are two types of capacity:

- **Representational capacity**
- **Effective capacity**

Capacity

Representational capacity

- The model specifies which family of functions the learning algorithm can choose from.

Effective capacity

- In practice, the learning algorithm does not actually find the best function, but merely one that significantly reduces the training error.

Relationship between model capacity & error

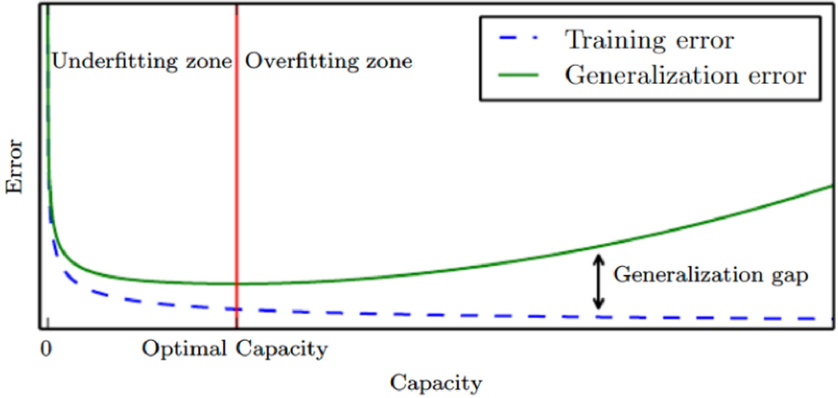
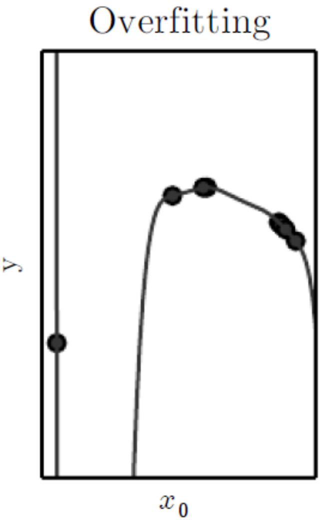


Figure: Relationship between model capacity and error.

Overfitting and Underfitting

Overfitting

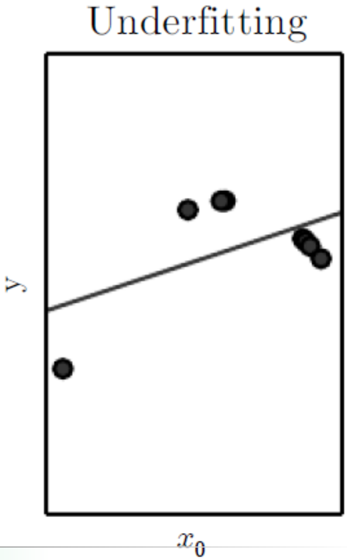
- When a model learns noise in the training .
- It negatively impacts the performance of the model on new data.



Overfitting and Underfitting

Underfitting

- It can neither fit the training data nor generalize to new data.
- Poor performance on the training and test data.

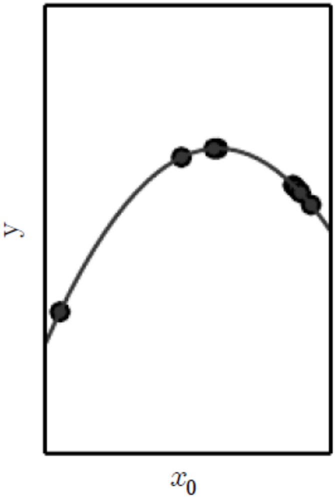


Good fit

Goal

Determining the sweet spot between underfitting and overfitting.

Sweet spot:
Sweet spot is the point just before the error on the test dataset starts to increase.



How To Limit Overfitting

Goal

We need to limit overfitting to have a better evaluation of our machine learning algorithm.

But How?

- There are few important techniques that we can use when evaluating deep learning algorithms to limit overfitting.
- Regularization is one of them.
- Different Regularization techniques are used for different DN models.

Regularization

Definition

Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error at the expenses of the training error.

Different Types of Regularization:

- Dataset Augmentation
- Parameter Norm Penalties
- Early Stopping
- Dropout

Dataset Augmentation

Background

The best way to make a machine learning model generalize better is to train it on more data.

How?

By creating fake data and adding it to the training set.

Applications:

- Speech recognition
- Object recognition

Dataset Augmentation

How Dataset Augmentation Works?

- Flipping (both vertically and horizontally)
- Rotating
- Zooming and scaling
- Cropping
- Translating (moving along the x or y axis)
- Adding Gaussian noise

Dataset Augmentation

Approaches for Dataset Augmentation

- **Offline dataset augmentation:**
  Transforms are applied to dataset before training
- **Online dataset augmentation:**
  Transforms are applied in real time as batches are passed into training.

Dataset Augmentation

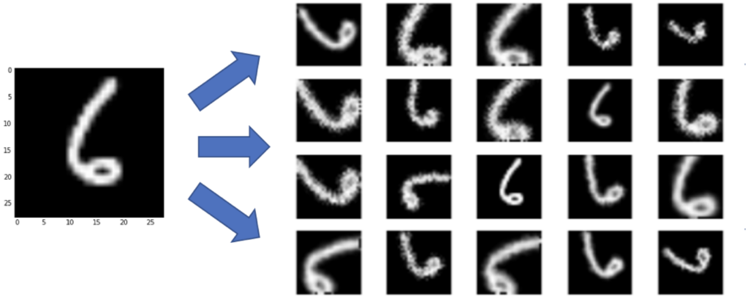


Figure: Visualization for Dataset Augmentation.

Parameter Norm Penalties Regularization:

Definition

$$\tilde{J}(\omega; X, Y) = J(\omega; X, Y) + \alpha\Omega(\theta)$$

- Here $\alpha \in [0, \infty)$ is a hyperparameter that weights the relative contribution of the norm penalty term, Ω, relative to the standard objective function J.

Parameter Norm Penalties Regularization

Differences Between L1 and L2 Regularization		
Topic	L1 Parameter Regularization	L2 Parameter Regularization
Cost	$\tilde{J}(\omega; X, Y) = J(\omega; X, Y) + \alpha \lVert\omega\rVert_1$	$\tilde{J}(\omega; X, Y) = J(\omega; X, Y) + \frac{\alpha}{2} \lVert\omega\rVert_2^2$
Computation	Computationally inefficient in non-sparse case	Computationally efficient due to having analytical solution
Solution type	Sparse Solutions	Non-Sparse Solutions
Solutions	L1 has multiple solutions	L2 has one solution
Features	Built in feature Selection	No Features selection

Parameter Norm Penalties Regularization

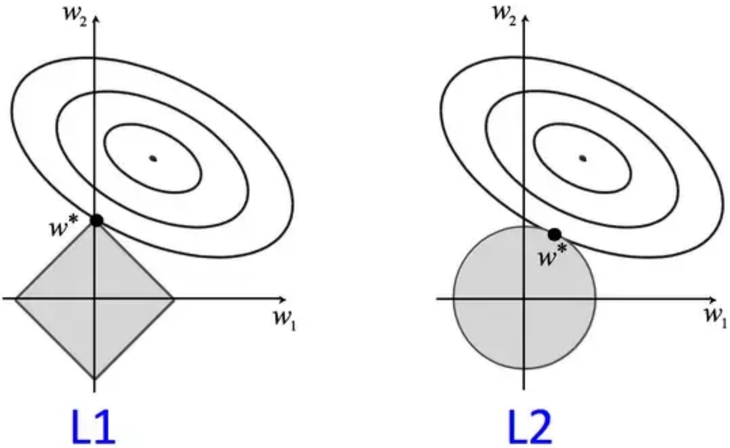
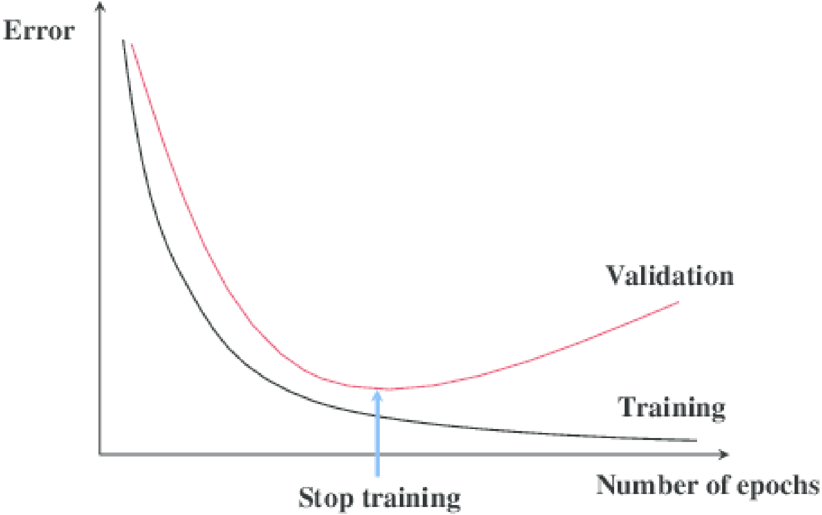


Figure: Visualization for L1 and L2 Regularization.

Early stopping



Dropout

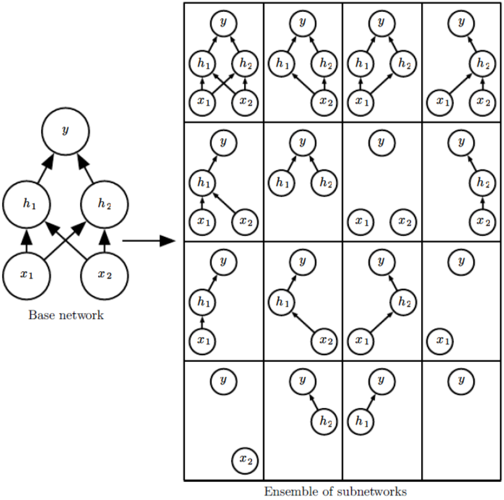


Figure: Visualization for Dropout Regularization.

Practical model

Topic:

Learning to Optimize: Training Deep Neural Networks for Wireless Resource Management.

Background:

The main idea is to treat a given resource optimization algorithm as a “black box”, and try to learn its input/output relation by using a deep neural network (DNN).

Motivation:

If a network with several layers can well approximate a given resource management algorithm, it will be economical in computation.

Goal:

The goal is the power allocation for each transmitter so that the weighted system throughput is maximized.

System Mathematical Model

$$\max_{p_1,\ldots,p_k} \sum_{k=1}^{K} \alpha_k \log(1 + \frac{|h_{kk}|^2 p_k}{\sum_{j \neq k} |h_{kj}|^2 p_j + \sigma_k^2}) \tag{1}$$

$$s.t. \quad 0 \leq p_k \leq P_{max} \qquad \forall k = 1, 2.., K$$

- Here P_{max} denotes the max power of each transmitter.
- $\alpha_k > 0$ are the weights.

The WMMSE Algorithm

The problem above can be solved by modified WMMSE algorithm.

$$\min_{(w_k,u_k,v_k)_{k=1}^{K}} \sum_{k=1}^{K} \alpha_k (w_k e_k - \log(w_k)) \quad (2)$$

$$s.t.\ \ 0 \leq v_k \leq \sqrt{P_k} \quad \forall k = 1;2..;K;$$

Here, e_k is defined as,

$$e_k = (u_k|h_{kk}|v_k - 1)^2 \sum_{j \neq k} (u_k|h_{kj}|v_j)^2 + \sigma_k^2 u_k^2 \quad (3)$$

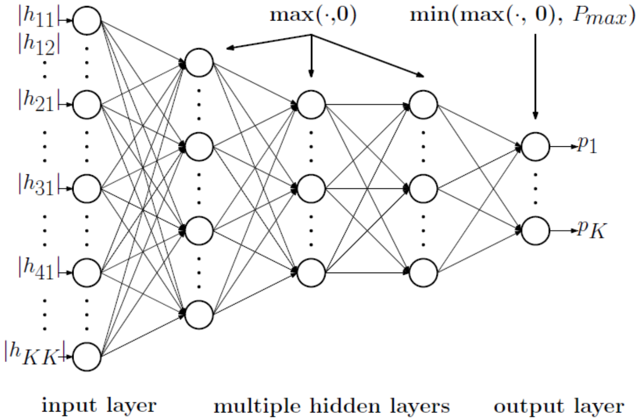
Data Generation

- First, the channel realizations $\{|h^i_{kj}|\}$ are generated following certain distributions.
- Then, , the corresponding optimized power vectors $\{p^i_k\}$ for each tuple $(P_{max}; \sigma_k; \{|h^i_{kj}|\})$ is generated by running WMMSE.
- Finally, the above process is repeated for multiple times to generate the entire training data set, as well as the validation data set.

The Proposed DNN Model

Network Structure	
Topic	Details
Input of the Network	$\lvert h_{kj} \rvert$
Output of the Network	p_k
Hidden Layers	3
Hidden Layers Neurons	360
Learning Rate	0.001
Hidden Layer Activation	ReLU
Output Layer Activation	$y = \min(\max(x;0);P_{max})$
Cost function	Mean squared error
Optimization algorithm	RMSprop algorithm

The Proposed DNN Model



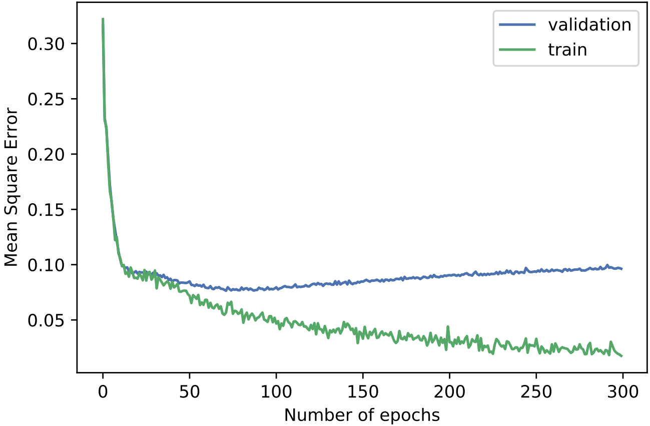
Our approach

Goal

To apply the following regularization techniques to the given model.

- L2 norm regularization
- Dropout regularization
- L2 + Dropout regularization

Simulated Results from the actual model



Simulated Results with L2 Regularization

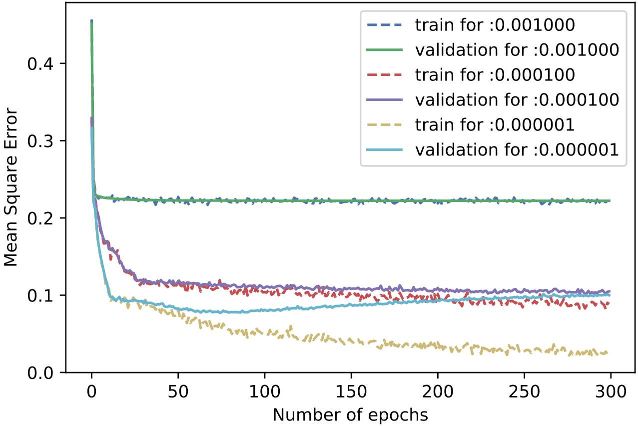
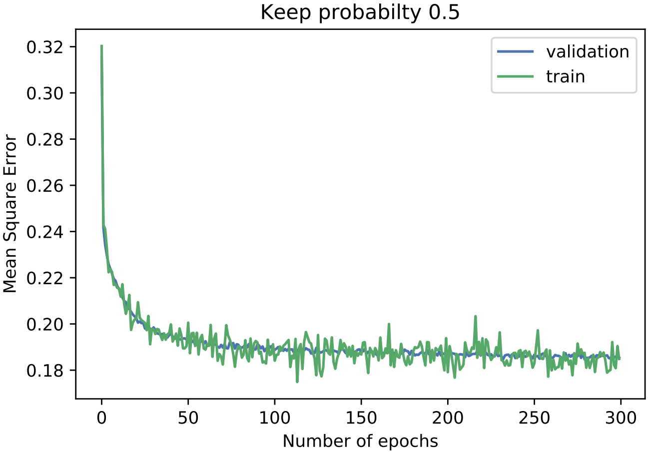
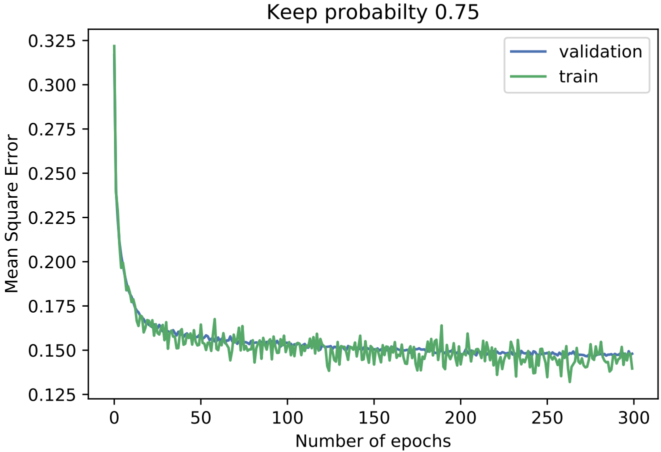


Figure: Mean square error for L2 regularization

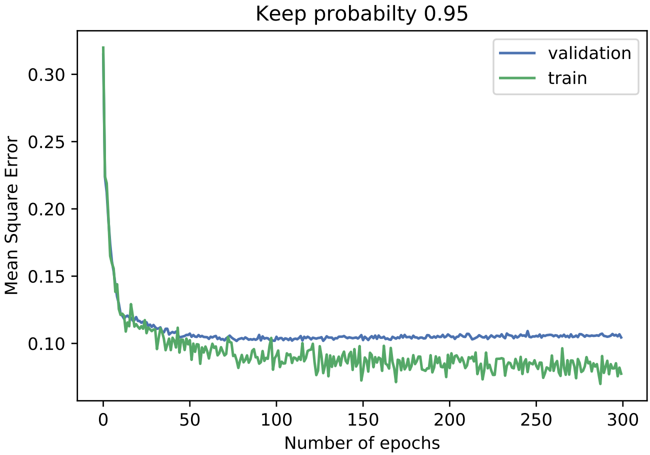
Simulated Results with Dropout Regularization



Simulated Results with Dropout Regularization



Simulated Results with Dropout Regularization



Results with Dropout+L2 Regularization

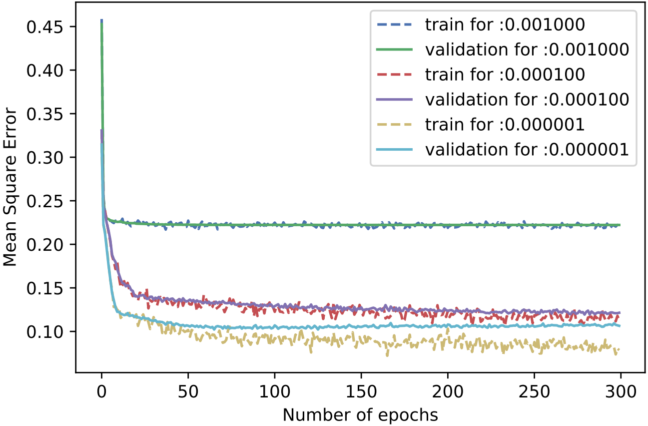


Figure: Mean square error for different L2 regularization constant with Dropout keep probability 0.95.

Conclusion

- We observed the effects of Regularization in our practical model.
- Early stopping could be the best solution for this particular model.

Thank you!

References I

- Ian Goodfellow, Yoshua Bengio, Aaron Courville
  Deep Learning.
  MIT Press, 2016.
- O. Simeone
  A Brief Introduction to Machine Learning for Engineers,2018.
- Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, Nicholas D. Sidiropoulos
  Learning to Optimize: Training Deep Neural Networks for Wireless Resource Management.
  IEEE Transactions on Signal Processing, vol. 66, no. 20, pp. 5438-5453, 15 Oct.15, 2018.

Regularizing a bigger model

- We want to make a bigger model with more neurons in every layer and apply same regularization methods as previous and observe the effects.
- Eventually we double the number of neurons in every hidden layer.

Simulated Results with the Bigger model

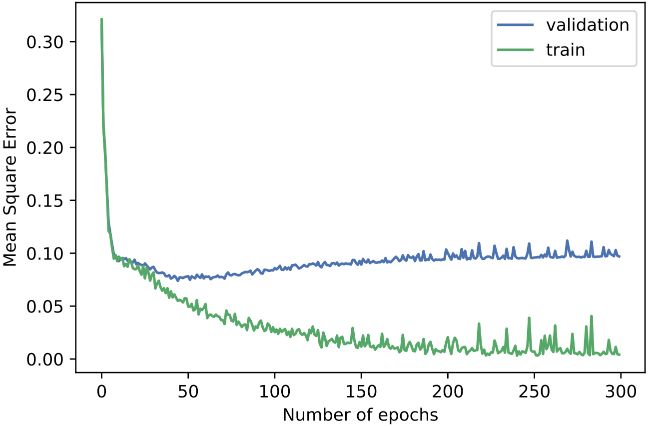


Figure: Mean square error for a Bigger model.

Simulated Results with Dropout Regularization

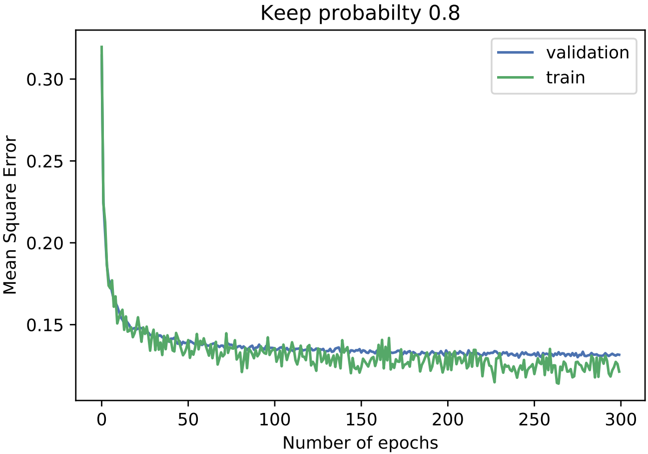


Figure: Mean square error for Dropout keep probabilty .8 .

Simulated Results with L2 Regularization

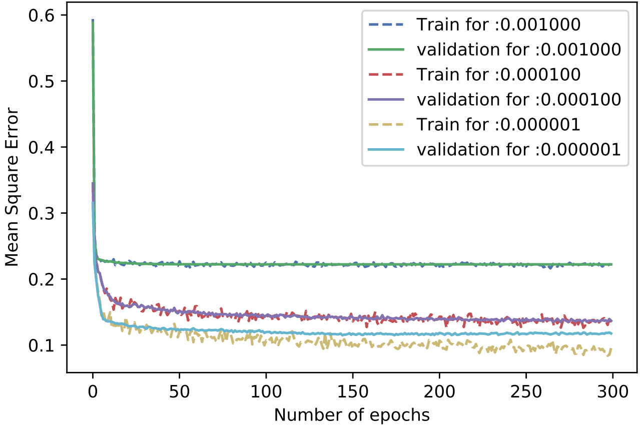


Figure: Mean square error for different L2 Regularization constant with Dropout keep probabilty .8.

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Regularization and Overfitting

Shrishti Saha Shetu

University of Erlangen-Nürnberg
Institute for Digital Communications

February 01, 2019