Early Stopping

Sargur N. Srihari
srihari@cedar.buffalo.edu

This is part of lecture slides on Deep Learning:
http://www.cedar.buffalo.edu/~srihari/CSE676

Regularization Strategies

1. Parameter Norm Penalties
2. Norm Penalties as Constrained Optimization
3. Regularization and Under-constrained Problems
4. Data Set Augmentation
5. Noise Robustness
6. Semi-supervised learning
7. Multi-task learning
8. Early Stopping
9. Parameter tying and parameter sharing
10. Sparse representations
11. Bagging and other ensemble methods
12. Dropout
13. Adversarial training
14. Tangent methods

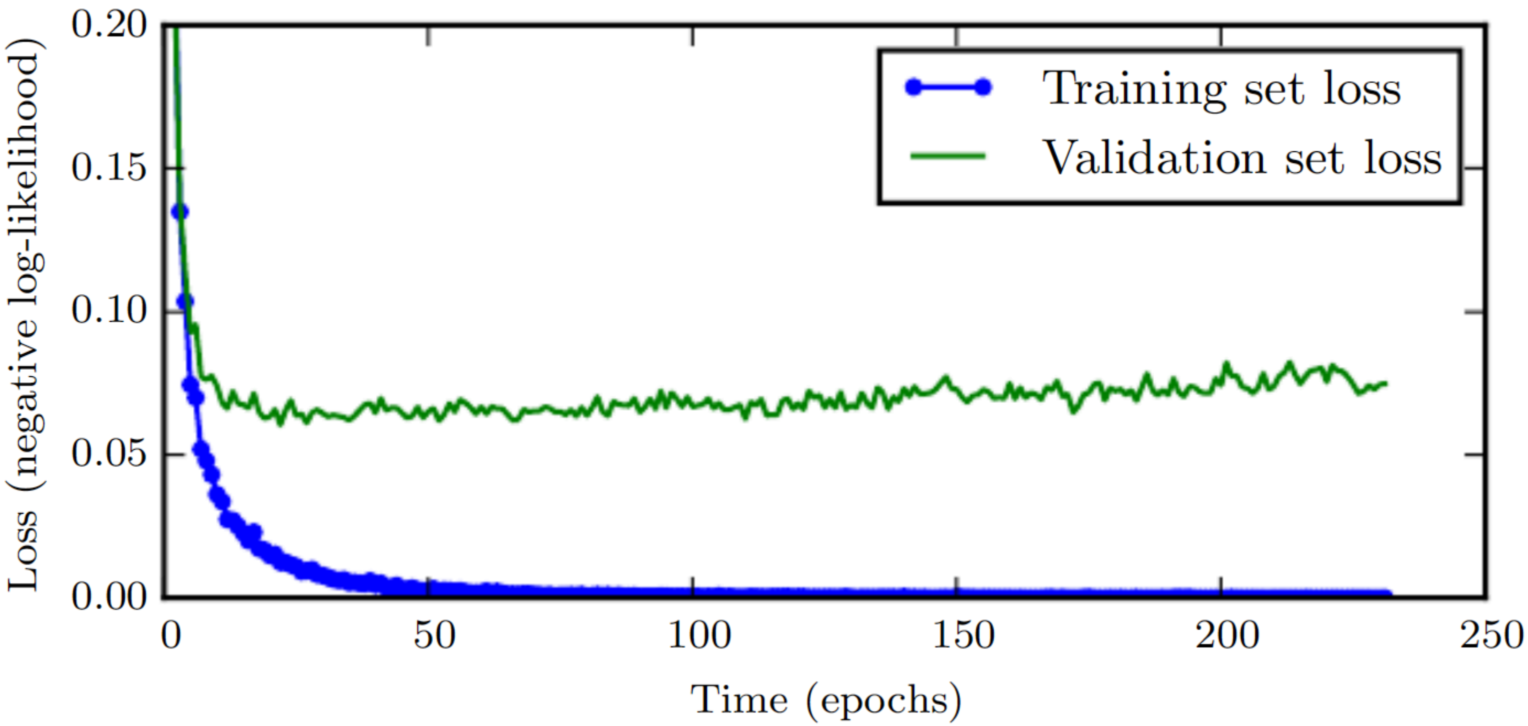
Topics in Early Stopping

1. Learning Curves
2. Early Stopping Meta-Algorithm
3. Early Stopping vs L^2 Regularization

Increase in validation set error

- When training large models with sufficient representational capacity to overfit the task, training error decreases steadily over time, but validation set error begins to rise again
- An example of this behavior is shown next

Learning Curves



Shows how negative log-likelihood loss changes over time (indicated as no. of Training iterations over the data set, or epochs).

In this example, we train a maxout network on MNIST (maxout generalizes RELU further)

Training objective decreases consistently over time, but validation set average Loss eventually begins to increase again forming an asymmetric U-shaped curve

Saving parameters

- We can thus obtain a model with better validation set error (and thus better test error) by returning to the parameter setting at the point of time with the lowest validation set error
- Every time the error on the validation set improves, we store a copy of the model parameters.
- When the training algorithm terminates, we return these parameters, rather than the latest set

Early stopping meta algorithm

Let n be the number of steps between evaluations.
Let p be the "patience," the number of times to observe worsening validation set error before giving up.
Let $\boldsymbol{\theta}_o$ be the initial parameters.
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_o$
$i \leftarrow 0$
$j \leftarrow 0$
$v \leftarrow \infty$
$\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
$i^* \leftarrow i$
while $j < p$ **do**
  Update $\boldsymbol{\theta}$ by running the training algorithm for n steps.
  $i \leftarrow i + n$
  $v' \leftarrow \text{ValidationSetError}(\boldsymbol{\theta})$
  if $v' < v$ **then**
    $j \leftarrow 0$
    $\boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}$
    $i^* \leftarrow i$
    $v \leftarrow v'$
  else
    $j \leftarrow j + 1$
  end if
end while
Best parameters are $\boldsymbol{\theta}^*$, best number of training steps is i^*

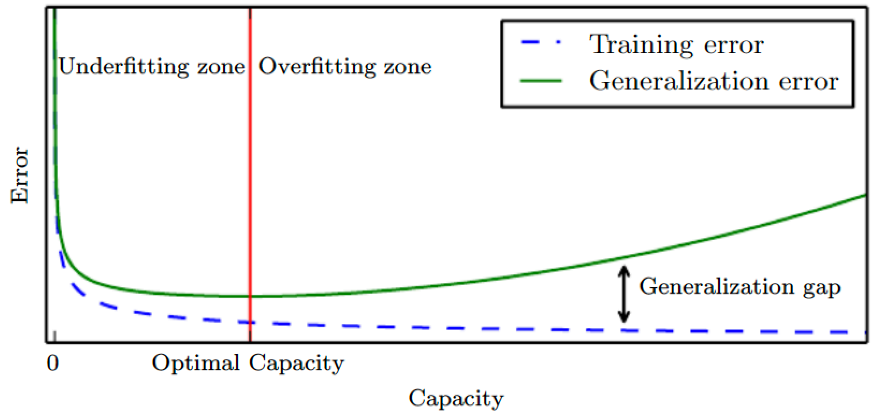
Algorithm determines the best amount of time to train. The meta algorithm is a general strategy that works well with a variety of training algorithms and ways of quantifying error on the validation set.

Strategy of Early Stopping

- The above strategy is known as Early Stopping
- It is the most common form of regularization in deep learning
- Its popularity is due to its effectiveness and its simplicity

Early Stopping as Hyperparameter Selection

- We can think of early stopping as a very efficient hyperparameter selection algorithm
  - In this view no. of training steps is just a hyperparameter
  - This hyperparameter has a U-shaped validation set performance curve
  - Most hyperparameters have such a U-shaped validation set performance curve, as seen below



In the case of early stopping we are controlling the effective capacity of the model by determining how many steps it can take to fit the training set

Costs of Early Stopping

- Cost of this hyperparameter is running validation evaluation periodically during training
  - Ideally done in parallel to training process on a separate machine
    - Separate CPU or GPU from main training process, Or
    - Using small validation set or validating set less frequently
- Need to maintain a copy of the best parameters
  - This cost is negligible because they can be stored on a slower, larger memory
    - E.g., training in GPU, but storing the optimal parameters in host memory or on a disk drive

Early Stopping as Regularization

- Early stopping is an unobtrusive form of regularization
- It requires almost no change to the underlying training procedure, the objective function, or the set of allowable parameter values
- So it is easy to use early stopping without damaging the learning dynamics
  - In contrast to weight decay, where we must be careful not to use too much weight decay
    - Otherwise we trap the network in a bad local minimum corresponding to pathologically small weights

Use of a second training step

- Early stopping requires a validation set
  - Thus some training data is not fed to the model
- To best exploit this extra data, one can perform extra training after the initial training with early stopping has completed
  - In the second extra training step, all the training data is included
- There are two basic strategies for the second training procedure

First Strategy for Retraining

- One strategy is to initialize the model again and retrain on all the data
- In the second training pass, we train for the same no. of steps as the erly stopping procedure determined was optimal in first pass
- Whether to retrain for the same no. of parameter updates or the same no of passes through the data set?
  - On the second round, each pass through dataset will require more parameter updates because dataset is bigger

First meta algorithm for retraining

A meta algorithm for using early stopping to determine how long to train, then retraining on all the data

Let $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ be the training set.
Split $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ into $(\boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{X}^{(\text{valid})})$ and $(\boldsymbol{y}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{valid})})$ respectively.
Run early stopping (algorithm 7.1) starting from random $\boldsymbol{\theta}$ using $\boldsymbol{X}^{(\text{subtrain})}$ and $\boldsymbol{y}^{(\text{subtrain})}$ for training data and $\boldsymbol{X}^{(\text{valid})}$ and $\boldsymbol{y}^{(\text{valid})}$ for validation data. This returns i^*, the optimal number of steps.
Set $\boldsymbol{\theta}$ to random values again.
Train on $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ for i^* steps.

Second strategy for retraining

- Keep all the parameters obtained from the first round of training and then *continue* training but now require using all the data
- We no longer have a guide for when to stop in terms of the no of steps
- Instead we monitor the average loss function on the validation set and continue training until it falls below the value of the training set objective of when early stopping halted

Second meta algorithm for retraining

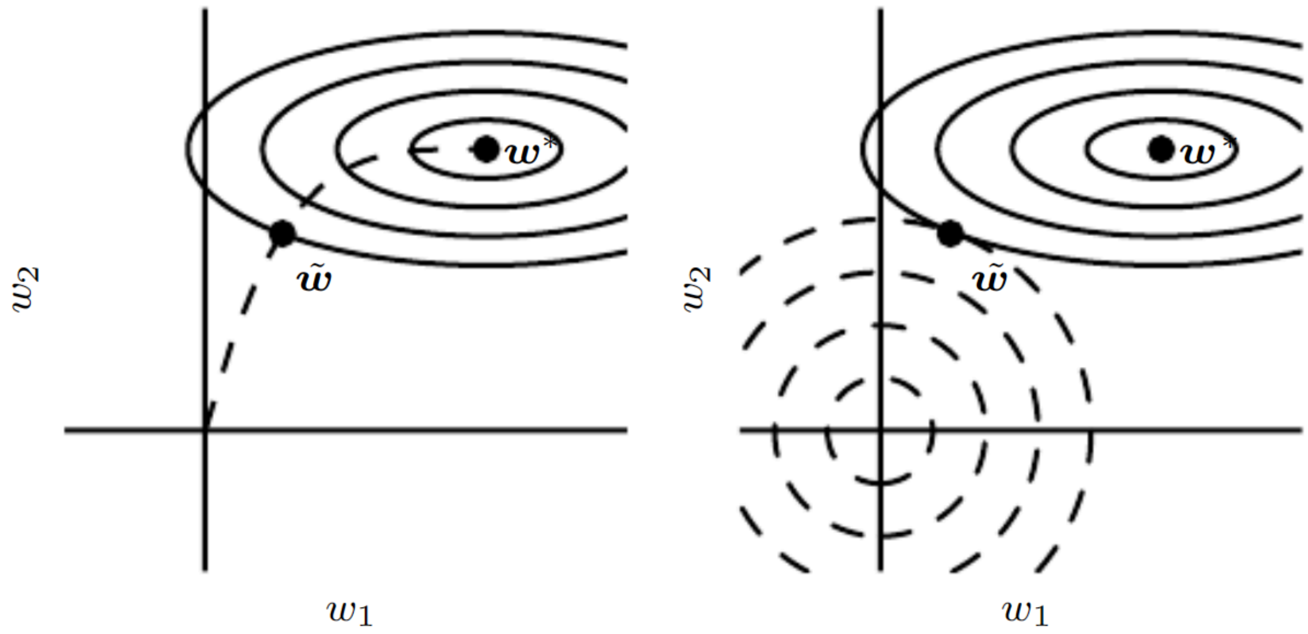
Meta-algorithm using early stopping to determine at what objective value we start to overfit, then continue training until that value is reached

Let $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ be the training set.
Split $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ into $(\boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{X}^{(\text{valid})})$ and $(\boldsymbol{y}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{valid})})$ respectively.
Run early stopping (algorithm 7.1) starting from random $\boldsymbol{\theta}$ using $\boldsymbol{X}^{(\text{subtrain})}$ and $\boldsymbol{y}^{(\text{subtrain})}$ for training data and $\boldsymbol{X}^{(\text{valid})}$ and $\boldsymbol{y}^{(\text{valid})}$ for validation data. This updates $\boldsymbol{\theta}$.
$\epsilon \leftarrow J(\boldsymbol{\theta}, \boldsymbol{X}^{(\text{subtrain})}, \boldsymbol{y}^{(\text{subtrain})})$
while $J(\boldsymbol{\theta}, \boldsymbol{X}^{(\text{valid})}, \boldsymbol{y}^{(\text{valid})}) > \epsilon$ **do**
  Train on $\boldsymbol{X}^{(\text{train})}$ and $\boldsymbol{y}^{(\text{train})}$ for n steps.
end while

Early stopping as a regularizer

- So far we have stated that early stopping is a regularization strategy
  - But supported the claim only by showing learning curves where the validation set error has a U-shaped curve
- What is the actual mechanism by which early stopping regularizes the model?
  - Early stopping has the effect of restricting the optimizing procedure to a relatively small volume of parameter space in the neighborhood of the initial parameters $\boldsymbol{\theta}_0$

Early Stopping vs L^2 regularization



- Two weights, Solid contour lines: contours of negative log-likelihood
- Left: dashed lines indicates trajectory of SGD. Rather than stopping at point w^* that minimizes cost, early stopping results in an earlier point in trajectory
- Right: dashed circles indicate contours of L^2 penalty which causes the minimum of the total cost to lie nearer the origin than the minimum of the the unregularized cost