Comprehensive Review: Issues in Training Neural Networks

DSC 255 - Machine Learning Fundamentals

Contents

1 Overview

Neural network training involves minimizing a highly non-convex loss function over potentially millions of parameters. This introduces several challenges in both generalization and optimization. This review outlines common techniques developed to address these issues.

2 Improving Generalization

Motivation

Neural networks have vast representational capacity. Without proper techniques, they risk overfitting the training data and failing to generalize to unseen examples.

2.1 Early Stopping

- Maintain a validation set separate from the training set.
- Track validation error periodically during training (e.g., every 100 steps).
- If validation error increases, revert to the best earlier model checkpoint.
- Intuition: Stop training before overfitting begins.

2.2 Dropout

- During training, randomly deactivate (drop out) each hidden unit with probability 0.5.
- This forces the network to develop redundant representations and prevents over-reliance on specific nodes.
- At test time, use the full network without dropout.
- Analogy: Similar in spirit to random forests and ensemble averaging.

3 Facilitating Optimization

3.1 Batch Normalization

- During training, the distribution of inputs to a hidden layer can shift (internal covariate shift).
- For each mini-batch and each feature x_i in a layer:
  - Compute batch mean μ_i and variance v_i
  - Normalize:
  $$x'_i = \frac{x_i - \mu_i}{\sqrt{v_i + \epsilon}}$$
- Effect: stabilizes and accelerates learning by reducing internal distribution shifts.

4 Variants of Gradient Descent

4.1 Standard SGD Update

$$\theta_{t+1} = \theta_t - \eta_t \cdot g_t$$

where $g_t = \nabla_\theta \ell(x_t, y_t; \theta_t)$ is the gradient of the loss at time t.

4.2 Momentum

$$\begin{aligned} v_t &= \mu v_{t-1} + \eta_t g_t \\ \theta_{t+1} &= \theta_t - v_t \end{aligned}$$

- Accumulates gradients over time to smooth updates.
- Helps escape shallow local minima or ravines.

4.3 AdaGrad

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \frac{\eta}{\sqrt{\sum_{s=1}^{t} (g_j^{(s)})^2 + \epsilon}} \cdot g_j^{(t)}$$

- Maintains per-parameter learning rates.
- Parameters with large historical gradients get smaller step sizes.
- Encourages learning-rate adaptation across sparse vs. dense features.

4.4 Other Methods

Popular alternatives:

- RMSProp
- Adam (Adaptive Moment Estimation)

These methods combine elements of momentum and per-parameter scaling.

5 Summary

- Training deep networks is difficult due to non-convexity and high variance.
- Techniques like early stopping and dropout improve generalization.
- Batch normalization and adaptive gradient methods facilitate efficient optimization.
- Neural net performance depends not only on architecture but also on regularization and optimizer choice.