TTIC 31230, Fundamentals of Deep Learning

David McAllester, April 2017

Over-Fitting and Regularization

Over Fitting

If we have more parameters than data then we can fit any set of labels.

> "Our experiments establish that state-of-the-art convolutional networks for image classification trained with stochastic gradient methods easily fit a random labeling of the training data."

Rethinking Generalization, Zhang et al., ICLR 2017

Train Data, Development Data and Test Data

Data is typically divided into **a training set**, **a development set** and **a test set** each drawn IID from the population.

A learning algorithm optimizes training loss.

One then optimizes algorithm design (and hyper-parameters) on the development set. (graduate student descent).

Ultimate performance should be done on a test set not used for development. Test data is often withheld from developers.

Loss Vs. Error Rate

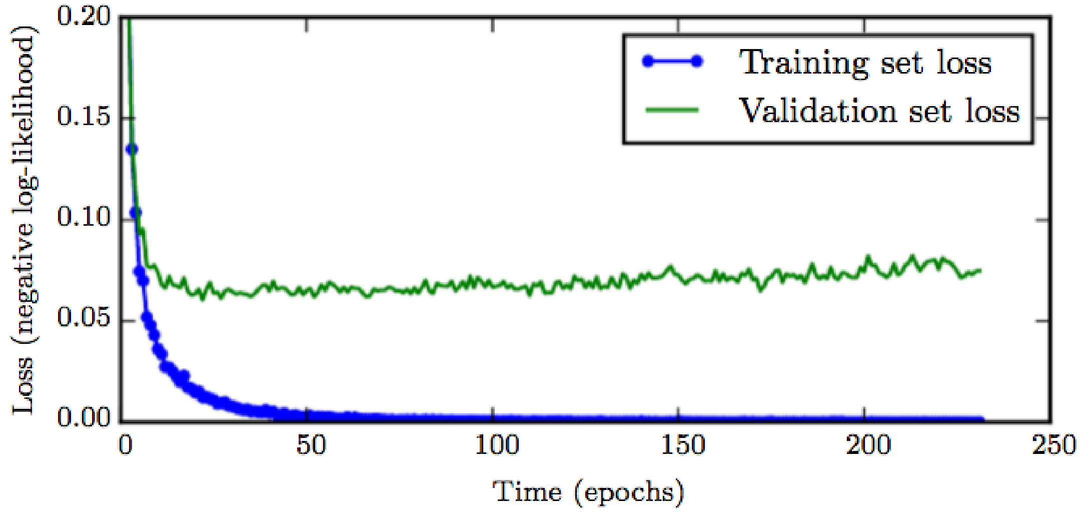
While training (gradient descent) is generally done on cross entropy loss, performance is often judged by other measures such as classification error rate.

The term “loss” is often used as a synonym for cross entropy loss.

Hence one often reports both “loss” and “error rate”.

Note that classification error rate is not differentiable.

Early Stopping

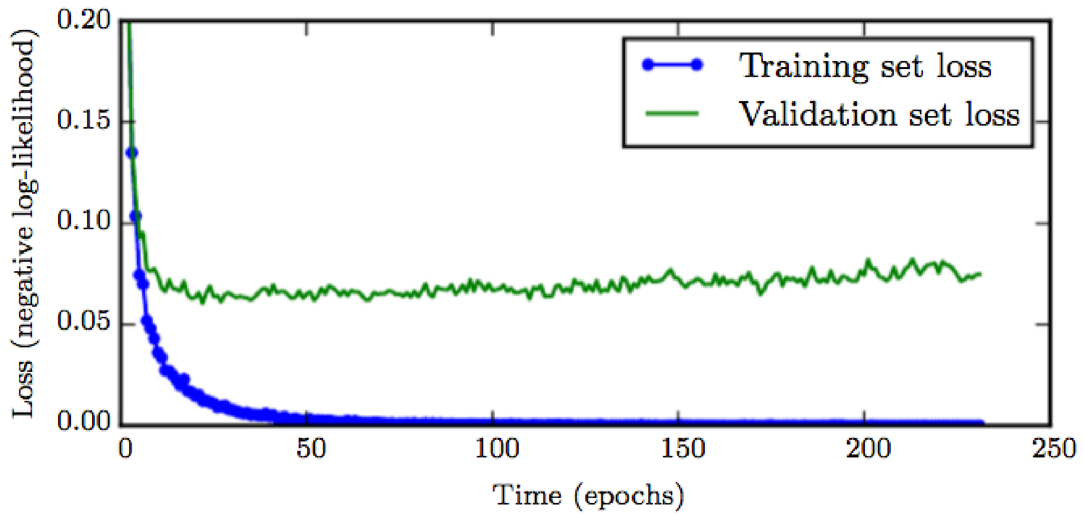


[Goodfellow et al.]

During SGD one should be tracking validation loss.

A typical rule is to stop when the validation loss has not set a new record low for some period.

Early stopping on Random Labels



[Goodfellow et al.]

For random labels the optimal score function is the constant zero which, for binary classification, has a loss of $\ln 2 \approx .69$.

If the validation data is random then the validation loss should start to climb immediately.

PAC-Bayesian Generalization Guarantees

It is possible to prove formal generalization guarantees without looking at the validation or test data.

To do this we assume a prior distribution on models.

Intuitively, for any prior (true or not) selected before seeing the data, one can prove that any model with high posterior probability must generalize well.

But there is no guarantee that any model will have sufficiently large posterior probability.

PAC-Bayesian Generalization Guarantees

Consider an arbitrary prior probability distribution over an arbitrary discrete class $\mathcal{H}$ of "hypotheses" (or models).

Consider a population distribution over pairs (x, y) with $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Consider a loss function $\mathcal{L}(h, x, y)$ such that for any $h \in \mathcal{H}$ and pair (x, y) we have $\mathcal{L}(h, x, y) \in (0, L_{\max})$.

For example $(h, x, y) = \min(L_{\max},\ -\ln\ P(y|h, x))$.

PAC-Bayesian Generalization Guarantees

$$\mathcal{L}(h) \;=\; E_{(x,y)\sim\mathrm{Pop}}\,\mathcal{L}(h,x,y)$$

$$\hat{\mathcal{L}}(h) \;=\; E_{(x,y)\sim\mathrm{Train}}\,\mathcal{L}(h,x,y)$$

Theorem: With probability at least $1-\delta$ over an I.I.D. draw of training data from the population the following holds *simultaneously* for all $h \in \mathcal{H}$.

$$\mathcal{L}(h) \leq \frac{10}{9}\left(\hat{\mathcal{L}}(h) + \frac{5L_{\max}}{N}\left(\ln\frac{1}{P(h)} + \ln\frac{1}{\delta}\right)\right)$$

A Model Compression Guarantee

Let $|\Phi|$ be the number of bits used to represent Φ under some fixed compression scheme.

Let $P(\Phi) = 2^{-|\Phi|}$

$$\mathcal{L}(\Phi) \leq \frac{10}{9}\left(\hat{\mathcal{L}}(\Phi) + \frac{5L_{\max}}{N}\left((\ln 2)|\Phi| + \ln\frac{1}{\delta}\right)\right)$$

A Weight Norm Guarantee

$$P(w) = \mathcal{N}(0,\sigma)^d$$

$$L(\Theta) = E_{\epsilon \sim \mathcal{N}(0,\sigma)^d}\ \mathcal{L}(\Theta + \epsilon)$$

$$\hat{L}(\Theta) = E_{\epsilon \sim \mathcal{N}(0,\sigma)^d}\ \hat{\mathcal{L}}(\Theta + \epsilon)$$

$$L(\Theta) \leq \frac{10}{9}\left(\hat{L}(\Theta) + \frac{5L_{\max}}{N}\left(\frac{||\Theta||^2}{2\sigma^2} + \ln\frac{1}{\delta}\right)\right)$$

L_2 Regularization (a.k.a. Weight Decay or Shrinkage)

$$\Phi^* = \operatorname*{argmin}_{\Phi} \; E_{(x,y)\sim \mathrm{Train}} \; \mathcal{L}(\Phi, x, y) + \frac{1}{2}\lambda||\Phi||^2$$

$$\nabla_\Phi \; \left(E_{(x,y)\sim \mathrm{Batch}} \; \mathcal{L}(\Phi, x, y) + \frac{1}{2}\lambda||\Phi||^2\right)$$

$$= \; g(\Phi) + \lambda\Phi \quad \text{giving} \;\; \Phi^* = \frac{g(\Phi^*)}{\lambda}$$

$$\Phi_{t+1} \; = \; \Phi_t - \eta\hat{g}_t - \eta\lambda\Phi_t$$

Standard Parameterization

The update

$$\Phi_{t+1} = \Phi_t - \eta \hat{g}_t - \eta \lambda \Phi_t$$

is typically parameterized as

$$\Phi_{t+1} = \Phi_t - \eta \hat{g}_t - \gamma \Phi_t$$

where γ is the weight decay parameter.

However, the η, λ parameterization may be less "entangled" in hyper-parameter search than the η, γ parameterization.

The Gradient Smoothing Parameterization

Setting $\eta = \gamma/\lambda$ we get

$$\begin{aligned} \Phi_{t+1} &= \Phi_t - \frac{\gamma}{\lambda}\hat{g}_t - \gamma\Phi_t \\ &= (1-\gamma)\,\Phi_t - \gamma\left(\frac{\hat{g}_t}{\lambda}\right) \\ \Phi^* &= \frac{g}{\lambda} \end{aligned}$$

Again, different parameterizations will have different degrees of entanglement between the parameters and reflect different semantic interpretations.

Shrinkage meets Early Stopping

Early stopping can limit $||\Phi||$ — growing a large $||\Phi||$ can take a long time.

Early stopping seems more related to limiting $||\Phi - \Phi_{\mathrm{init}}||$

Theoretical guarantees work for $||\Phi - \Phi_{\mathrm{init}}||^2$ just as well as for $||\Phi||^2$.

This suggests replacing L_2 regularization with

$$\Phi^* = \operatorname*{argmin}_{\Phi} \mathcal{L}(\Phi) + \frac{1}{2}\lambda||\Phi - \Phi_{\mathrm{init}}||^2$$

Shrinkage meets Batch Scaling

We have some in-house evidence that shinkage is important in batch scaling.

Batch scaling with shrikage seems better behaved than batch scaling without shrikage.

?????

L_1 Regularization and Sparse Weights

$$p(\Phi) \propto e^{-||\Phi||_1} \qquad ||\Phi||_1 = \sum_i |\Phi_i|$$

$$\Phi^* = \operatorname*{argmin}_\Phi \quad \hat{\mathcal{L}}(\Phi) \; + \; \lambda ||\Phi||_1$$

$$\begin{aligned} \Phi \;\; \texttt{-=} \;\; & \eta \nabla_\Phi \, \ell_{\mathrm{train}}(\Phi) \\ \Phi_i \;\; \texttt{-=} \;\; & \eta\lambda \; \mathrm{sign}(\Phi_i) \qquad \text{(shrinkage)} \end{aligned}$$

At equilibrium (sparsity is difficult to achieve with SGD)

$$\begin{aligned} \Phi_i &= 0 & & \text{if } |\partial\ell/\partial\Phi_i| < \lambda \\ \partial\ell/\partial\Phi_i &= -\lambda \mathrm{sign}(\Phi_i) & & \text{otherwise} \end{aligned}$$

Ensembles

Train several models $\mathrm{Ens} = (\Phi_1, \ldots, \Phi_k)$ from different initializations and/or under different meta parameters.

We define the ensemble model by

$$P_{\mathrm{Ens}}(y|x) = \frac{1}{k}\sum_{j=1}^{k} P_{\Phi_i}(y|x)$$

Ensemble models almost always perform better than any single model.

We will explore some reasons for this.

Ensembles Under Cross Entropy Loss

For log loss we average the probability vectors.

$$P(y|x) = \frac{1}{k} \sum_i \; P_i(y|x)$$

$-\log P$ is a convex function of P. For any convex $\ell(P)$ Jensen's inequality states that

$$\ell\left(\frac{1}{k}\sum_i P_i\right) \leq \frac{1}{k}\sum_i \ell(P_i)$$

This implies that the loss of the average model cannot be worse (can only be better) than the average loss of the models.

Implicit Regularization

Any stochastic learning algorithm, such as SGD, determines a stochastic mapping from training data to models.

The algorithm can implicitly incorporate a preference or bias for models.

For example, solving linear least squares regression with SGD maintains the invariant that Φ is a linear combination of training vectors.

It is not hard to show that SGD finds the zero training error solution minimizing $||\Phi||$.

So solving least squares regression by SGD has an implicit weight norm regularization.

An Implicit Regularization Generalization Guarantee

let $A(\mathrm{Train})$ be the distribution on models defined by running the stochastic algorithm A on training data Train.

We let $\overline{A}$ be the distribution on models defined first drawing Train and then drawing a model from $A(\mathrm{Train})$.

$$\overline{A}(\Phi) = E_{\mathrm{Train}\sim\mathrm{Pop}}\; A(\mathrm{Train})(\Phi)$$

$$L_A(\mathrm{Train}) = E_{\Phi\sim A(\mathrm{Train})};E_{(x,y)\sim\mathrm{Pop}}\;\mathcal{L}(\Phi,x,y)$$

$$\hat{L}_A(\mathrm{Train}) = E_{\Phi\sim A(\mathrm{Train})}\;E_{(x,y)\sim\mathrm{Train}}\;\mathcal{L}(\Phi,x,y)$$

An Implicit Regularization Generalization Guarantee

For any given learning algorithm $\mathcal{A}$ and we have the following with probability at least $1-\delta$ over the draw of Train.

$$L(\mathrm{Train}) \leq \frac{10}{9}\left(\hat{L}(\mathrm{Train}) + \frac{5L_{\max}}{N}\left(KL(A(\mathrm{Train}),\ \overline{A}) + \ln\frac{1}{\delta}\right)\right)$$

Dropout

Dropout can be viewed as an ensemble method.

To draw a model from the ensemble we randomly select a mask μ with

$$\begin{cases} \mu_i = 0 \text{ with probability } \alpha \\ \\ \mu_i = 1 \text{ with probability } 1 - \alpha \end{cases}$$

Then we use the model $(\Phi,\ \mu)$ with weight layers defined by

$$y_i = \text{Relu}\left(\sum_j W_{i,j}\, \mu_j x_j \right)$$

A Dropout Bound

$$
\begin{aligned}
KL(Q_{\alpha,\Phi},\ Q_{\alpha,0}) &= E_{\mu\sim P_\alpha,\epsilon\sim\mathcal{N}(0,1)^d}\ \ln\frac{P_\alpha(\mu)e^{-\frac{1}{2}||\mu\odot\epsilon||^2}}{P_\alpha(\mu)e^{-\frac{1}{2}||\mu\odot(\Phi+\epsilon)||^2}} \\
&= E_{\mu\sim P_\alpha}\ \frac{1}{2}||\mu\odot\Phi||^2 \\
&= \frac{1-\alpha}{2}||\Phi||^2
\end{aligned}
$$

$$
L(Q_{\alpha,\Phi}) \leq \frac{1}{1-\frac{1}{2\lambda}}\left(\hat{L}(Q_{\alpha,\Phi}) + \frac{\lambda L_{\max}}{N}\left(\frac{1-\alpha}{2}||\Phi||^2 + \ln\frac{1}{\delta}\right)\right)
$$

Dropout Training

Repeat:

- Select a random dropout mask μ
- Φ `-=` $\nabla_\Phi \, \ell(\Phi, \mu)$

Backpropagation must use the same mask μ used in the forward computation.

Test Time Scaling

At train time we have

$$y_i = \text{Relu}\left(\sum_j W_{i,j}\ \mu_j x_j\right)$$

At test time we have

$$y_i = \text{Relu}\left((1-\alpha)\ \sum_j W_{i,j}\ x_j\right)$$

At test time we use the "average network".

Dropout for Least Squares Regression

Consider simple least square regression

$$\begin{aligned}\Phi^* &= \operatorname*{argmin}_{\Phi} \;\; \mathrm{E}_{(x,y)}\, E_\mu \left(y - \Phi \cdot (\mu \odot x)\right)^2 \\ &= \mathrm{E}\left[(\mu \odot x)(\mu \odot x)^\top\right]^{-1} \mathrm{E}\left[y(\mu \odot x)\right] \\ &= \operatorname*{argmin}_{\Phi} \;\; \mathrm{E}_{(x,y)}(y - (1-\alpha)\Phi \cdot x)^2 + \sum_i \frac{1}{2}(\alpha - \alpha^2)\mathrm{E}\left[x_i^2\right]\Phi_i^2\end{aligned}$$

In this case dropout is equivalent to a form of L_2 regularization — see Wager et al. (2013).

Proof of the Discrete PAC-Bayes Bound

$$\mathcal{L}(h) = E_{(x,y)\sim \mathrm{Pop}}\, \mathcal{L}(h,x,y)$$

$$\hat{\mathcal{L}}(h) = E_{(x,y)\sim \mathrm{Train}}\, \mathcal{L}(h,x,y)$$

Proof

Consider $L_{\max} = 1$ and define $\epsilon(h)$ by

$$\epsilon(h) = \sqrt{\frac{2\mathcal{L}(h)\left(\ln \frac{1}{P(h)} + \ln \frac{1}{\delta}\right)}{N}}.$$

By the relative Chernoof bound we have

$$P_{\text{Train}\sim\text{Pop}}\left(\hat{\mathcal{L}}(h) \leq \mathcal{L}(h) - \epsilon(h)\right) \leq e^{-N\frac{\epsilon(h)^2}{2\mathcal{L}(h)}} = \delta P(h).$$

Proof

$$P_{\mathrm{Train}\sim\mathrm{Pop}}\left(\hat{\mathcal{L}}(h) \leq \mathcal{L}(h) - \epsilon(h)\right) \leq \delta P(h).$$

$$P_{\mathrm{Train}\sim\mathrm{Pop}}\left(\exists h\ \hat{\mathcal{L}}(h) \leq \mathcal{L}(h) - \epsilon(h)\right) \leq \sum_h \delta P(h) = \delta$$

$$P_{\mathrm{Train}\sim\mathrm{Pop}}\left(\forall h\ \mathcal{L}(h) \leq \hat{\mathcal{L}}(h) + \epsilon(h)\right) \geq 1 - \delta$$

Proof

$$\mathcal{L}(h) \leq \widehat{\mathcal{L}}(h) + \sqrt{\mathcal{L}(h)\left(\frac{2\left(\ln\frac{1}{P(h)} + \ln\frac{1}{\delta}\right)}{N}\right)}$$

using

$$\sqrt{ab} = \inf_{\lambda>0}\ \frac{a}{2\lambda} + \frac{\lambda b}{2}$$

we get

$$\mathcal{L}(h) \leq \widehat{\mathcal{L}}(h) + \frac{\mathcal{L}(h)}{2\lambda} + \frac{\lambda\left(\ln\frac{1}{P(h)} + \ln\frac{1}{\delta}\right)}{N}$$

Proof

$$\mathcal{L}(h) \leq \widehat{\mathcal{L}}(h) + \frac{\mathcal{L}(h)}{2\lambda} + \frac{\lambda\left(\ln\frac{1}{P(h)} + \ln\frac{1}{\delta}\right)}{N}$$

Solving for $\mathcal{L}(h)$ yields

$$\mathcal{L}(h) \leq \frac{1}{1-\frac{1}{2\lambda}}\left(\hat{\mathcal{L}}(h) + \frac{\lambda}{N}\left(\ln\frac{1}{P(h)} + \ln\frac{1}{\delta}\right)\right)$$

Setting $\lambda = 5$ and rescaling the loss gives the version on earlier slides.

END