Stability and SGD

ATML Track1

Rik Sarkar

Today

- Regularization
- Leave one out and Uniform Stability
- Relation between regularization, stability and generalization
- SGD
- Theoretical properties
- Variants

Regularization

- Instead of the pure loss, minimize loss with a regularization term:

$$\underset{\mathbf{w}}{\operatorname{argmin}} \left(L_S(\mathbf{w}) + R(\mathbf{w})\right)$$

- Commonly used: $R(\boldsymbol{w}) = \lambda \left|\left|\boldsymbol{w}\right|\right|^2$
    - Called Tikhonov regularization

- $R(\boldsymbol{w}) = \lambda||\boldsymbol{w}||^2$
  - Is 2λ-strongly convex

- If $L_S(w)$ is convex, then $L_S(w) + R(W)$ is 2-strongly convex

- Strong convexity implies stability

Stability

- Intuitively: A learning algorithm is stable if
  - A small change to training set does not cause a big change to the output (model or hypothesis)

- This is a desirable property because…

Stability

- Intuitively: A learning algorithm is stable if
  - A small change to training set does not cause a big change to the output (model or hypothesis)
- This is a desirable property because
  - It implies that it is not too sensitive to specific S. does not overfit
  - If we continue to use it, it will not abruptly change behavior as new data comes in

- Suppose in S, we replace z_i with $z' \sim \mathcal{D}$
- Let us write this as S^i

- A good algorithm A should have small value for
  - $|\ell(A(S^i), z_i) - \ell(A(S), z_i)|$

- The loss at z_i does not depend too much on it being in the sample

Stability definition

- Algorithm A is on-average-replace-one-stable with rate $\epsilon(m)$
- If
  - $\mathbb{E}_{S,z'\sim\mathcal{D}^{m+1},i\sim U(m)}\left[\ell\left(A\left(S^i\right),z_i\right)-\ell(A(S),z_i)\right]\leq\epsilon(m)$
  - Expectation is over $S\sim\mathcal{D}^m, z'\in\mathcal{D}$ and i selected uniformly from $[1,m]$

Tikhonov regularization creates stability

- For $A(S) = \mathrm{argmin}_w \left(L_S(w) + \lambda \left|\left|\boldsymbol{w}\right|\right|^2\right)$

- $\ell\left(A\left(S^i\right), z_i\right) - \ell\left(A(S), z_i\right) \leq \frac{2\rho^2}{\lambda m}$

- How does the stability change with training data size?

- Why do we need strong convexity?

Stability definition and result

- Algorithm A is on-average-replace-one-stable with rate $\epsilon(m)$
- If
  - $\mathbb{E}\left[\ell\left(A\left(S^i\right), z_i\right) - \ell(A(S), z_i)\right] \leq \epsilon(m)$

- Theorem:
  - $\mathbb{E}\left[L_{\mathcal{D}}\left(A(S)\right) - L_S\left(A(S)\right)\right] = \mathbb{E}\left[\ell\left(A\left(S^i\right), z_i\right) - \ell(A(S), z_i)\right]$

  - Stability implies generalisation

Uniform Stability

- Suppose we get S^i by replacing one element z_i at position i of S with a new element z_i'
- And suppose that $z \in \mathcal{D}$ is some possible input element
- As before $A(S)$ refers to the model that algorithm A computes using S
- We can write the loss on z as $\ell(A(S), z)$

- Algorithm A is ϵ_{stab}-uniformly stable if
  - $\text{Sup}_{z \in \mathcal{D}}\left[E_A \ell\left(A(S^i), z\right) - E_A \ell(A(S), z)\right] \leq \epsilon_{\text{stab}}$
- E_A means expectation taken over all possible random behaviour of A
  - E.g. randomization in SGD

Stability implies generalization

- Theorem:
- If Algorithm A is ϵ-uniformly stable then
  - $E_S E_A \ell(A(S), \mathcal{D}) \leq E_S E_A \ell(A(S), S) + \epsilon_{\text{stab}}$
  - True loss $\leq$ Training loss $+\epsilon_{\text{stab}}$

- Regularization creates strong convexity
- Strong convexity implies stability
- Stability implies generalization

Stochastic Gradient descent

- The problem with gradient descent
  - Computing the average loss L over all of S is expensive

- Idea:
  - We just need a good enough gradient vector
    - Does not need to be perfect. As long as it takes us in about the right direction
  - This can be obtained by a sample of S – much more efficient
  - A large enough sample of S will take us in almost the same direction as S
  - A small sample will take us in the right direction *in expectation*
    - Repeated use of small samples should work well.

SGD

- Start with $\boldsymbol{w}^0$ initialised randomly (Within some bound B, e.g. unit ball)
- At every step t :
  - Sample a data point $z = (x, y) \in S$
  - Update $\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \nabla \ell(\boldsymbol{w}^t, z)$
    - (Move in the direction that loss ℓ decreases fastest With a step factor of η)

- After T steps, output the average vector $\bar{\boldsymbol{w}} = \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{w}^t$
- Other version: output final vector $\boldsymbol{w_T}$
- Note that this time the function of interest is loss on the single data point z

Convergence

- For B bounded and ρ Lipschitz, convex loss
- Setting $\eta = \sqrt{\frac{B^2}{\rho^2 T}}$
- For expected loss $\mathbb{E}[L(\bar{\boldsymbol{w}})]$:
  - $\mathbb{E}[L(\bar{\boldsymbol{w}})] - L(\boldsymbol{w}^*) \leq \frac{B\rho}{\sqrt{T}}$

- Therefore, to achieve $\mathbb{E}[L(\bar{\boldsymbol{w}})] - L(\boldsymbol{w}^*) \leq \epsilon$, the number of rounds is
- $T \geq \frac{B^2 \rho^2}{\epsilon^2}$

Guarantee for strongly convex loss

- Assuming λ-strongly convex, ρ-Lipschitz loss,
- An SGD with $\eta_t = \frac{1}{\lambda t}$
- $\mathbb{E}[L(\bar{\boldsymbol{w}})] - L(\boldsymbol{w}^*) \leq \frac{\rho^2}{2\lambda T}(1 + \log T)$

Stability

- Suppose we write α_t for the step size at round t
- For convex loss that is ρ-Lipschitz and β-smooth
- If we run SGD with step size $\alpha_t \leq 2/\beta$
- Then the algorithm is uniformly stable with
    - $\epsilon_{\text{stab}} \leq \frac{2\rho^2}{m} \sum_{t=1}^{T} \alpha_t$

Common modifications

- Mini-batch SGD
  - Instead of a single $z \in S$ each round, use a small batch (e.g. 128)
  - Mini batch (larger sample) gives more accurate gradient – faster arrival at min
- Run in epochs. In each epoch
  - Order the data points in a random permutation
  - And iterate through the permutation instead of a random sample each time
- Momentum
  - Adjust gradient vector based on recent movement
  - Avoids excessive impact from current batch
- Adjusting learning rates
- Moving average of gradients
- ADAM: momentum and moving average of gradients

Various forms of implicit regularizations

- Momentum
- Early stopping
- Small parameters at initialization
- Data augmentation
- In neural networks
  - Dropout
  - Batch normalization