Regularization

Professor Marie Roch

Regularization

Goal: Reduce generalization error

Strategies:

• Constraints on parameter values
  e.g. try to keep parameters from being too small/large

• Preferences for simpler models

• Guidance for underspecified problems

• Combine multiple hypotheses (ensemble methods)
Parameter norm penalties

Attempt to limit capacity

\[
\tilde{J}(\theta \mid X, y) = J(\theta \mid X, y) + \alpha \Omega(\theta)
\]

- \(\alpha \in [0, \infty)\) is user settable
- Common to use a L\(p\) norm for \(\Omega(\Theta)\)
- Model \(\Theta\) comprised of weights \(w\);
  \(\tilde{J}(w \mid X, y)\) is equivalent

Remember \(L^p\), or \(\|\| = \left(\sum_{i} |x_i|^p\right)^{1/p}\) (Goodfellow et al. 2.5)

Parameter norm penalties

- Common to ignore bias
  - only shifts position
  - penalizing frequently results in underfitting
- Separate \(\alpha\) per layer is possible, but complicates finding the right set of hyperparameters. Reasonable to just use one globally.
Parameter norm penalties

We will discuss two:

• L2 – Causes weights to get smaller
  – Shrinkage proportional to weight
  – AKA “weight decay”

• L1 – Makes weight vector “sparse”
  – Pulls weights towards zero by constant factors

\[ \Omega(\theta) = \alpha w^T w \]

Also known as ridge regression & Tikhonov regularization
L² penalty

- Consider weight update

\[
w \leftarrow w - \epsilon \nabla_w \tilde{J}(\theta | X, y)
\]

\[
w - \epsilon \nabla_w \tilde{J}(\theta | X, y) =
\]

\[
w - \epsilon (\nabla_w J(\theta | X, y) + \alpha w) =
\]

\[
w - \epsilon \alpha w - \epsilon \nabla_w J(\theta | X, y) =
\]

\[
(1 - \epsilon \alpha)w - \epsilon \nabla_w J(\theta | X, y)
\]

\[
w \leftarrow (1 - \epsilon \alpha)w - \epsilon \nabla_w J(\theta | X, y)
\]

Tends to shrink weights (with appropriate \(\alpha\))

---

L² penalty

- For a quadratic penalty (e.g. regression), this has the affect of scaling directions by the eigen-values of the Hessian matrix

oh, oh… rabbit hole
Hessian matrix – H

\( g : \mathbb{R}^n \rightarrow \mathbb{R}^n \)

\[
\begin{bmatrix}
\frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \cdots & \frac{\partial y_1}{\partial x_M} \\
\frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_2}{\partial x_M} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial y_N}{\partial x_1} & \frac{\partial y_N}{\partial x_2} & \cdots & \frac{\partial y_N}{\partial x_M}
\end{bmatrix}
\]

Jacobian (saw earlier)

\[
\begin{bmatrix}
\frac{\partial^2 y_1}{\partial x_1 \partial x_1} & \frac{\partial^2 y_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 y_1}{\partial x_1 \partial x_M} \\
\frac{\partial^2 y_2}{\partial x_1 \partial x_1} & \frac{\partial^2 y_2}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 y_2}{\partial x_1 \partial x_M} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 y_N}{\partial x_1 \partial x_1} & \frac{\partial^2 y_N}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 y_N}{\partial x_1 \partial x_M}
\end{bmatrix}
\]

Hessian is 2\textsuperscript{nd} partial derivative

Goodall et al. 4.3.1

Hessian matrix

So why do we care…

- Sometimes used in gradient descent
- Provides an explanation for what’s happening with L\textsuperscript{2} penalty.
L² penalty

Consider approximation near optimal point without regularization: \( w^* \)

\[
\hat{J}(w) = J(w^*) + \frac{1}{2} (w - w^*)^T H(w - w^*)
\]

- \( w \in \mathbb{R}^2 \)
- \( w^* \) optimal w/o penalty
- \( w_2 \) has more affect on \( J(w) \)
- \( \bar{w} \) equilibrium between loss and penalty

\( w_1 \) less sensitive than \( w_2 \)
eigen vectors of Hessian capture directions,
eigen values capture magnitudes

Fig. 7.1 Goodall et al.
L1 penalty

- L1 penalty $\Omega(\theta) = \sum_i |w_i|$

\[
\tilde{J}(\theta \mid X, y) = J(\theta \mid X, y) + \alpha \Omega(\theta)
\]
\[
= J(\theta \mid X, y) + \alpha \sum_i |w_i|
\]

- Gradient

\[
\nabla_w \tilde{J}(\theta \mid X, y) = \nabla_w J(\theta \mid X, y) + \alpha \text{sign}(w)
\]

L1 penalty

- As weights are pulled towards zero, fewer will be active.
- Leads to more zeros, or a sparse representation
- Can be thought of as a type of feature selection and is used in one popular algorithm (LASSO).
Keras and L1/L2 penalties

Penalties are specified when constructing layers, example

```python
from keras import regularizers
# other imports...

Dense(N, kernel_regularizer=regularizers.l2(α), ...)
OR
Dense(N, kernel_regularizer=regularizers.l1(α), ...)
```

Starting point for α? Perhaps 0.01

Dataset augmentation

- We can improve classifiers by increasing training data.
- Data are expensive (most of the time)
- Solution: Very fake data...
Dataset augmentation

• Primary application: classification tasks
• Basic idea
  – Transform inputs slightly
  – Frequently used in image processing
  – Transforms such as rotation, scale, shift

Dataset augmentation for audio

• Can be a bit harder
• Some strategies
  – vocal tract length perturbation (Jaitly & Hinton 2013)
  – enhancements, e.g. adding noise (Prisyach et al. 2015)
  (small perturbations of inputs can be shown to be equivalent to $L^p$ penalties on weights)
Noise robustness

- In addition to input perturbation, adding noise to weights can be effective
- Interpretations for weight perturbation:
  - Weight values have a distribution and we are drawing from these (e.g. Bayesian view)
  - With MSE cost functions, can show that $\hat{J}(w|X, y)$ contains additional term $\|\nabla_w \hat{y}(x)\|^2$, meaning weight changes have small influence on output; that is minima in flat valleys in loss space

Label Smoothing

- Inject noise at output targets
- Maximizing log $P(y|x)$ can lead to overfitting.
- Common to inject noise onto output target.
- You have already done this: softmax