Riemannian approach to batch normalization

Minhyung Cho — mhyung.cho@gmail.com
Jaehyung Lee — jaehyung.lee@kaist.ac.kr

Overview

Batch normalization

$$\mathrm{BN}(z)=\frac{z-E[z]}{\sqrt{Var[z]}}=\frac{w^T(x-E[x])}{\sqrt{w^TR_{xx}w}}=\frac{u^T(x-E[x])}{\sqrt{u^TR_{xx}u}}$$

where $u=w/|w|$ and R_{xx} is the covariance matrix of x

- forward pass is scale invariant
  $$\mathrm{BN}(w^Tx)=\mathrm{BN}(u^Tx)$$
- backward pass is scale invariant
  $$\frac{\partial \mathrm{BN}(w^Tx)}{\partial x}=\frac{\partial \mathrm{BN}(u^Tx)}{\partial x}$$
- weight update is **not** scale invariant
  $$\frac{\partial \mathrm{BN}(z)}{\partial w}=\frac{1}{|w|}\frac{\partial \mathrm{BN}(z)}{\partial u}$$

Things we observe …

- Two networks, with the same forward pass but different weight scaling, may fall into different local minima
- L_2 regularization of the weights is known to be indispensable for the performance, but |w| does not affect the output of the network. Why is it important then?
- The scale of |w| affects the learning rate, so L_2 regularization functions as a learning rate control unexpectedly

What we did …

- Give scale invariance to the weight update step

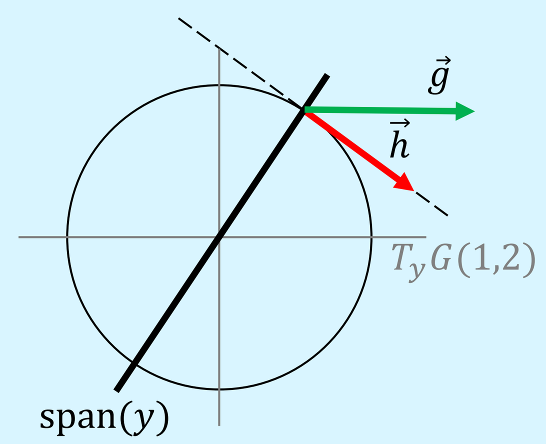
How?

- Utilize the geometry of the space of scale invariant vectors
- Derive learning rules for this space
- Derive a new regularization method in this space

Operators on a Grassmann manifold G(1,n)

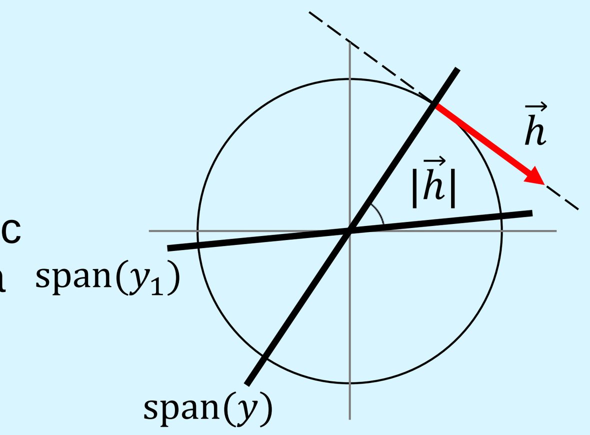
Gradient of a function

- grad $f(y)\in T_y\mathcal{M}$
- Gradient of f on a manifold is a tangent vector to the manifold
- grad $f=g-(y^Tg)y$ where $g_i=\partial f/\partial y_i$



Exponential map

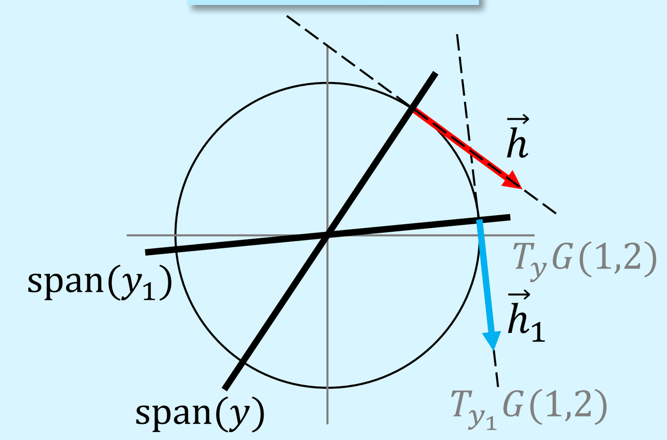
- $\exp_y(h)$ where $y\in\mathcal{M}, h\in T_y\mathcal{M}$
- Move y along a unique geodesic on $\mathcal{M}$, with initial velocity h, in a unit time
- $\exp_y(h)=y\cos|h|+\frac{h}{|h|}\sin|h|$



$y_1=\exp_y(h)$

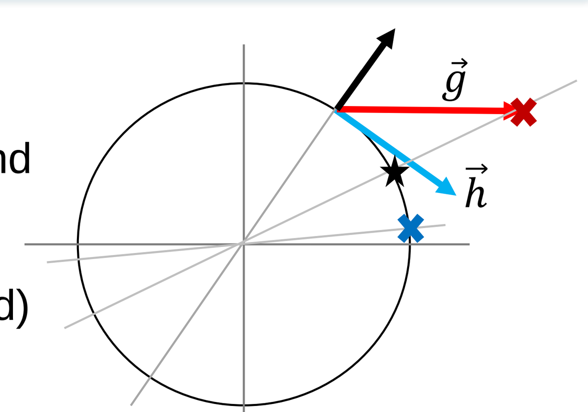
Parallel translation

- $\mathrm{pt}_y(\Delta;h)$ where $y\in\mathcal{M}$ and $\Delta,h\in T_y\mathcal{M}$
- Parallel translate Δ along the geodesic with the initial velocity h in a unit time
- $\mathrm{pt}_y(h;h)=h\cos|h|-y|h|\sin|h|$



$h_1=\mathrm{pt}_y(h;h), |\vec{h}|=|\vec{h}_1|$

- $\mathbb{R}^2$ vs G(1,2)
  - $\mathbb{R}^2$: Only the tangent direction $\vec{h}$ contributes to minimizing cost, and the normal direction disturbs the learning rate
  - G(1,2) : Move the point by |h| (rad)



- $y=\exp_y\left(2\pi\cdot\frac{h}{|h|}\right)$
  - It returns to the original point after moving by 2π → gradient clipping is necessary

Proposed algorithms

We derive iterative algorithms to solve the unconstrained optimization on G(1,n) :

$$\min_{y\in G(1,n)} f(y)$$

- Optimizations algorithms in Euclidean space can be easily extended to those on manifolds by properly using the operators defined on the manifolds
- A weight vector is **a point** on the manifold, so we can move it by the **exponential map**
- Gradient is **a tangent vector** to the manifold, so the momentum (accumulation of gradient) can be moved by the **parallel translation**

Algorithm 1) SGD with momentum on G(1,n) : SGD-G

Require: learning rate η, momentum coefficient γ, norm_threshold ν
Initialize $y_0\in\mathbb{R}^{n\times1}$ with a random unit vector
Initialize $\tau_0\in\mathbb{R}^{n\times1}$ with a zero vector
for $t=1,\cdots,T$

$g\leftarrow\partial f(y_{t-1})/\partial y$	Run a backward pass to obtain g
$h\leftarrow g-(y_{t-1}^Tg)y_{t-1}$	Project g onto the tangent space at y_{t-1}
$\hat{h}\leftarrow\mathrm{norm\_clip}(h,\nu)^\dagger$	Clip the norm of the gradient at ν
$d\leftarrow\gamma\tau_{t-1}-\eta\hat{h}$	Update delta with momentum
$y_t\leftarrow\exp_{y_{t-1}}(d)$	Move to the new position by the exponential map
$\tau_t\leftarrow\mathrm{pt}_{y_{t-1}}(d)$	Move the momentum by the parallel translation

$^\dagger\mathrm{norm\_clip}(h,\nu)=\nu\cdot h/|h|$ if $|h|>\nu$, else h

- To apply to a neural network with BN layers:

for W ={weight matrices such that W^Tx is an input to a BN layer}
  Let W be an $n\times p$ matrix
  for $i=1,\cdots,p$
    $m\leftarrow m+1$
    Assign a column vector w_i in W to $y_m\in G(1,n)$
  Assign remaining parameters to $v\in\mathbb{R}^l$
for $t=1,\cdots,T$
  Run a forward pass to calculate the loss L
  Run a backward pass to obtain $\frac{\partial L}{\partial y_i}$ for $i=1,\cdots,m$ and $\frac{\partial L}{\partial v}$
  for $i=1,\cdots,m$
    Update the point y_i by SGD-G or Adam-G
  Update v by conventional optimization algorithms (such as SGD)
For orthogonality regularization, replace L with $L+\sum_W L^O(\alpha,W)$

- The forward pass and backward pass remain unchanged
- Adam on G(1,n) : Adam-G
  - The adaptive step size is given to each weight vector rather than each parameter. In this way, the direction of the gradient is not corrupted, and the size of the step is adaptively controlled
  - Please refer to the paper for details
- Regularization on G(1,n) : $L^O(\alpha,Y)=\frac{\alpha}{2}\|Y^TY-I\|_F^2$

Experiments

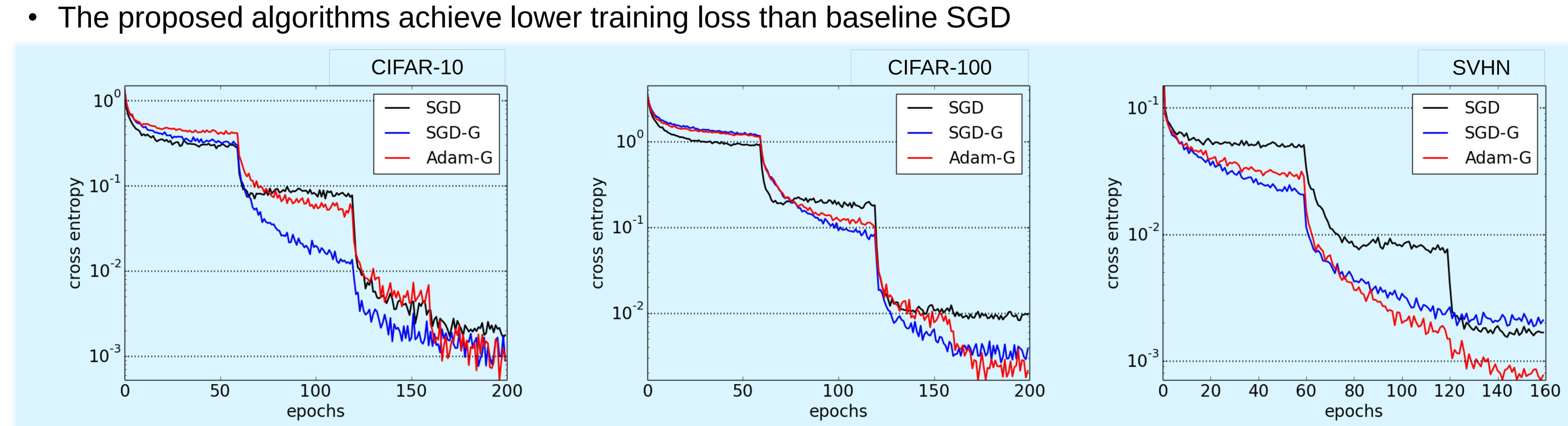
- Classification error rate on CIFAR (median of five runs): WRN-d-k denotes a wide residual network that has d convolutional layers and a widening factor k

Dataset	CIFAR-10			CIFAR-100		
Model	SGD	SGD-G	Adam-G	SGD	SGD-G	Adam-G
VGG-13	5.88	5.87	6.05	26.17	25.29	24.89
VGG-19	6.49	5.92	6.02	27.62	25.79	25.59
WRN-28-10	3.89	3.85	3.78	18.66	18.19	18.30
WRN-40-10	**3.72**	**3.72**	3.80	18.39	18.04	**17.85**

- Classification error rate on SVHN (median of five funs):

Dataset	SVHN		
Model	SGD	SGD-G	Adam-G
VGG-13	1.78	1.74	1.72
VGG-19	1.94	1.81	1.77
WRN-16-4	1.64	1.67	1.61
WRN-22-8	1.64	1.63	**1.55**

- The proposed algorithms suffer less from a plateau after each learning rate drop
- The proposed algorithms achieve lower training loss than baseline SGD



Source code for the experiments is available at https://github.com/MinhyungCho/riemannian-batch-normalization

Space of scale invariant vectors

We want a space where all the scaled versions of a vector collapse to a point

1) 1-d Grassmann manifold G(1,n)
   - x and y are equivalent if and only if $x=ay$ for $a\in\mathbb{R}\backslash\{0\}$
   - $g_g(\Delta_1,\Delta_2)=\Delta_1^T\Delta_2/y^Ty$

2) 1-d Stiefel manifold V(1,n)
   - equivalent to the unit sphere, $|x|=1$
   - $g_s(\Delta_1,\Delta_2)=\Delta_1^T\Delta_2$

If we choose a representation y with $y^Ty=1$, G(1,n) and V(1,n) offer the same metric and resulting operators

Riemannian BN