ASSIGNMENT 2: THEORY OF CNNS AND REGULARIZATION [IFT6135]

JOSEPH D. VIVIANO

1. Convolutions

$\boldsymbol{x} = (..., x_{-1}, x_0, x_1, x_2, x_3, x_4, ...) = (..., 0, 1, 2, 3, 4, 0, ...)$
$\boldsymbol{w} = (..., w_{-1}, w_0, w_1, w_2, w_3, w_4, ...) = (..., 0, 1, 0, 2, 0, 0, ...)$

$$[\boldsymbol{x} * \boldsymbol{w}](i) = \sum_{k=-\infty}^{\infty} x_k w_{i-k} \tag{1.1}$$

$[\boldsymbol{x} * \boldsymbol{w}](0) = x_0 w_0 = 1 \cdot 1 = 1$

$[\boldsymbol{x} * \boldsymbol{w}](1) = x_0 w_1 + x_1 w_0 = 1 \cdot 0 + 2 \cdot 1 = 2$

$[\boldsymbol{x} * \boldsymbol{w}](2) = x_0 w_2 + x_1 w_1 + x_2 w_0 = 1 \cdot 2 + 2 \cdot 0 + 3 \cdot 1 = 5$

$[\boldsymbol{x} * \boldsymbol{w}](3) = x_0 w_3 + x_1 w_2 + x_2 w_1 + x_3 w_0 = 1 \cdot 0 + 2 \cdot 2 + 3 \cdot 0 + 4 \cdot 1 = 8$

$[\boldsymbol{x} * \boldsymbol{w}](4) = x_2 w_2 + x_3 w_1 = 3 \cdot 2 + 4 \cdot 0 = 6$

$[\boldsymbol{x} * \boldsymbol{w}](5) = x_3 w_2 = 4 \cdot 2 = 8$

Let's $y_i = [\boldsymbol{x} * \boldsymbol{w}](i)$. Thus we have $\boldsymbol{y} = (1, 2, 5, 8, 6, 8)$ where the null elements are not shown.

2. Convolutional Neural Networks

(a) NB: zero padding in layer three, so size of layer 3 is $128 \times 6 \times 6$. The image is RGB (i.e., 3 channels), therefore the last layer is a fully connected layer of the size

$$3 \times 128 \times 6 \times 6 = 13824$$

.

(b) The last convolution has a kernel size 4×4 and there are 128 filters with 3 channels, so

$$n\_params = 4 \times 4 \times 128 \times 3 = 6144$$

.

Date: March 2018.

3. Kernel Configurations for CNNs

(a): ***i*** : input is $W_1 \times H_1$ and output is $W_2 \times H_2$. Kernel size is K, zero padding is P and stride is S. Therefore

$$W_2 = \frac{W_1 - K + 2P}{S} + 1, \tag{3.1}$$

plugging our numbers in, we get

$$32 = \frac{64 - 8 + 2P}{S} + 1$$

. Either $\boldsymbol{P = 3}$ and $\boldsymbol{S = 2}$ would produce a proper convolution.

ii : Dilatation size is D,

$$W_2 = \frac{W_1 - K + 2P + (W_1 - 1)D}{S} + 1 \tag{3.2}$$

So plugging in,

$$32 = \frac{64 - K + 2P + 63.6}{2} + 1.$$

If we set $\boldsymbol{K = 400}$ and $\boldsymbol{P = 10}$, then our convolution operation works.

(b): If the kernel size of the pooling layer is $\boldsymbol{K = 4 \times 4}$ with no overlap, and the stride size is $\boldsymbol{S = 4}$, the pooling operation works.

(c): $K = 8$, $W_1 = 32$ and $S = 4$, we plug them in and presto

$$W_2 = \frac{32 - K}{4} + 1 = 7.$$

The output is $\boldsymbol{7 \times 7}$.

(d): ***i*** $W_2 = 4$, $W_1 = 8$ and $P = 0$, plugging in, we get

$$4 = \frac{8 - K + 0}{S} + 1.$$

Therefore $\boldsymbol{K = 2}$ and $\boldsymbol{S = 2}$ are appropriate.

ii : $W_2 = 4$, $W_1 = 8$, $P = 2$ and $D = 1$. Plugging in, we get

$$4 = \frac{8 - K + 4 + 7}{S} + 1.$$

So $\boldsymbol{K = 13}$ and $\boldsymbol{S = 2}$ are appropriate.

iii :

$$4 = \frac{8 - K + 2}{S} + 1,$$

so $\boldsymbol{K = 4}$ and $\boldsymbol{S = 2}$ are appropriate.

4. Dropout as Weight Decay

(a) Let $\tilde{X} = X \odot \boldsymbol{\delta}$ where $\boldsymbol{\delta} = (\delta_1, ..., \delta_n)$,

$$\delta_i = \begin{cases} \mathbf{0}, & p \\ \mathbf{1}, & 1-p, \end{cases} \tag{4.1}$$

and $\mathbf{0}$ and $\mathbf{1}$ are vecteur of dimension $1 \times d$ of elements 0 and 1 respectively.

(b) Let L_{MSE} be the cost function. The general formula for the cost is

$$L_{MSE}(w) = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2 \tag{4.2}$$

where $\hat{y}^{(i)}$ is the the prediction and if we add dropout we have

$$L_{MSE}(w) = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)}\delta_i w - y^{(i)})^2 \tag{4.3}$$

(c) In this part, we will assume that the probability of dropping a input unit is $1-p$. The expected value of the prediction is

$$\mathbb{E}[\hat{y}^{(i)}] = \mathbb{E}[X\delta_i w] = Xwp. \tag{4.4}$$

This mean that the expected value of the mean square error when using dropout is obtain by replacing the parameters vector w by wp. If we use matrices to express the expected cost we have

$$\begin{aligned} L_{MSE}(w) =& (Xwp - y)^\top (Xwp - y) \\ =& p^2 w^\top X^\top X w - 2pw^\top X^\top y + y^\top y \end{aligned} \tag{4.5}$$

Now we take the derivatives with respect to w

$$\frac{\partial}{\partial w} L_{MSE}(w) = 2p^2 X^\top X w - 2pX^\top y \tag{4.6}$$

And making the dervatives equal to zero

$$\begin{aligned} 2p^2 X^\top X w^* - 2pX^\top y =& 0 \\ X^\top X w^* p =& X^\top y \\ w^* p =& (X^\top X)^{-1} X^\top y \end{aligned} \tag{4.7}$$

5. Dropout as a Geometric Ensemble

Consider the case of a single linear layer model with a softmax output. Prove that weight scaling by 0.5 corresponds exactly to the inference of a conditional probability distribution proportional to a geometric mean over all dropout masks.

First, observe the single linear layer with softmax output with n input variables represented by the vector v with dropout mask d:

$$P(y = \mathrm{y}|v; d) = \mathbf{softmax}\left(W^T(d \odot v) + b\right)_y \tag{5.1}$$

and the ensemble conditional probability distribution which represents the geometric mean over all dropout masks:

$$p_{ens}(y = \mathrm{y}|v; d) \propto \left(\prod_{i=1}^{N} \hat{y}_v^{(i)}\right)^{\frac{1}{N}}. \tag{5.2}$$

Aren't they nice? Recall the alternative formulation of the softmax:

$$\mathbf{softmax}_i = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}} \tag{5.3}$$

Which we now rewrite, subbing in our vector representation of the softmax and replacing e^x with $exp(x)$:

$$\mathbf{softmax}_y = \frac{exp\left(W_y^T(d \odot v) + b\right)}{\sum_{k=1}^{K} exp\left(W_{y'}^T(d \odot v) + b\right)} \tag{5.4}$$

Now we show that the ensemble predictor is defined by re-normalizing the geometric mean over all the individual ensemble members' predictions:

$$P_{ens}(y = \mathrm{y}|v) = \frac{\tilde{P}_{ens}(y = \mathrm{y}|v)}{\sum y'\tilde{P}_{ens}(y = \mathrm{y'}|v)} \tag{5.5}$$

Where each $\tilde{P}_{ens}$ is the geometric mean over all dropout masks for a single y:

$$\tilde{P}_{ens}(y = \mathrm{y}|v) = 2^n\sqrt{\prod_{d\in\{0,1\}^n} P(y = \mathrm{y}|v; d)}. \tag{5.6}$$

Now we simply sub in our definition of $softmax$ for P:

$$\tilde{P}_{ens}(y = \mathrm{y}|v) = 2^n\sqrt{\prod_{d\in\{0,1\}^n} \frac{exp\left(W_y^T(d \odot v) + b\right)}{\sum_{k=1}^{K} exp\left(W_{y'}^T(d \odot v) + b\right)}}. \tag{5.7}$$

Since the denominator is a constant under this normalization scheme we ignore it and simplify:

$$\tilde{P}_{ens}(y = \mathrm{y}|v) \propto 2^n \sqrt{\prod_{d\in\{0,1\}^n} exp\left(W_y^T(d\odot v) + b\right)} \tag{5.8}$$

We convert the product to the sum by taking *exp* of the entire equation:

$$\tilde{P}_{ens}(y = \mathrm{y}|v) \propto exp\left(\frac{1}{2^n}\sum_{d\in\{0,1\}^n} W_y^T(d\odot v) + b\right) \tag{5.9}$$

And finally the sum and exponent n cancel:

$$\tilde{P}_{ens}(y = \mathrm{y}|v) \propto exp\left(\frac{1}{2}W_y^T(d\odot v) + b\right) \tag{5.10}$$

Finally, we sub this back into our earlier formulation of the softmax to show that the weights W are scaled by $\frac{1}{2}$:

$$\mathbf{softmax}_y = \frac{exp\left(\frac{1}{2}W_y^T(d\odot v) + b\right)}{\sum_{k=1}^{K} exp\left(\frac{1}{2}W_{y'}^T(d\odot v) + b\right)} \tag{5.11}$$

Therefore, weight scaling by 0.5 is exactly equivilant to a conditional probability distribution proportional to a geometric mean over all dropout masks.

6. Normalization

(a) *Show batchnorm and weightnorm are the same when you only have one layer and input feature x.*

To normalize the minibatch of activations B, we do

$$B' = \frac{B-\mu}{\sigma} \tag{6.1}$$

Where μ is the mean of B, and σ is the standard deviation of B (with a small positive value added for numerical stability).
We can replace B with $w^\top x$, where w is our weight matrix to see:

$$B' = \frac{w^\top x}{\sqrt{\mathrm{Var}[w^\top x]}} - \frac{\mathbb{E}w^\top x}{\sqrt{\mathrm{Var}[w^\top x]}} \tag{6.2}$$

Now notice the following about the unit vector

$$\frac{u}{||u||} = \frac{w^\top}{||w||} \tag{6.3}$$

And

$$g = \frac{||w||_2}{\sqrt{\mathrm{Var}[w^\top x]}} \tag{6.4}$$

Therefore if we assume x and w are independent and that x has 0 mean:

$$B' = \frac{||w||_2}{\sqrt{\mathrm{Var} w^\top x}} \frac{w^\top}{||w||_2} x - \frac{\mathbb{E} w^\top x}{\sqrt{\mathrm{Var}[w^\top x]}} \tag{6.5}$$

$$B' = g \frac{u}{||u||} x - \frac{\mathbb{E} w^\top x}{\sqrt{\mathrm{Var}[w^\top x]}} \tag{6.6}$$

The expectation and standard deviation are constant under these conditions, so we get:

$$B' = g \frac{u}{||u||} x - c \tag{6.7}$$

But we can ignore c for this question.

(b) *Show the gradients of L with respect to u can be expressed as $sW^* \nabla_w L$.*

From above:

$$B' = g \frac{u}{||u||} x \tag{6.8}$$

With weightnorm, we explicity reparameterize the model to perform gradient descent in the new parameters g and u directly. By decoupling the norm of the weight vector g and it's direction $\frac{u}{||u||}$, we can speed up convergence dramatically.

If we differentiate through the above with respect to some new parameters v, we get:

$$\nabla_g L = \frac{\nabla_u L \cdot u}{||u||}, \nabla_v L = \frac{g}{||u||} \nabla_u L - \frac{g \nabla_g L}{||u||^2} u \tag{6.9}$$

Where $\nabla_u L$ is the gradient with respect to the weights.
Let's sub in $\nabla_g L$ into $\nabla_v L$ to get:

$$\nabla_v L = \frac{g}{||u||} \nabla_u L - \frac{g \frac{\nabla_u L \cdot u}{||u||}}{||u||^2} u \tag{6.10}$$

Or,

$$\nabla_v L = \frac{g}{||u||} \nabla_u L - \frac{g \nabla_u L}{||u||^3} u^\top u \tag{6.11}$$

This leads us to the forumulation:

$$\nabla_v L = \frac{g}{||u||} M_u \nabla_u L \tag{6.12}$$

where,

$$M_u = Id - \frac{u^\top u}{||u||^2} \tag{6.13}$$

where where M_u is a projection matrix that projects onto the complement of the u vector, and Id is the identity matrix.

(c) *Explain a graph of different learning rates.*

Let λ be the learning rate. During learning we update u via at step k using $v_k \leftarrow v_k - \lambda \nabla_v L$.

As we said in the previous question, the matrix M_u project onto the complement of u. Therefore, $\nabla_v L$ is equal to a constant times M_u, i.e., $u \perp \lambda \nabla_v L$.

Since our update v is proportional to w, the update must be orthogonal to v and the norm increases by the Pythagorean theorem, which states that for any two orthogonal vectors v and v' the new weight vector must have the norm

$$||v'|| = \sqrt{||v||^2 + c^2||v||^2} \tag{6.14}$$

if

$$c = ||\lambda \nabla_v L|| / ||v|| \tag{6.15}$$

This tells us a few things. If the norm of the gradients is small, $\sqrt{1+c^2}$ is close to 1 and the norm of v stops increasing. As the norm of the gradients grow, the norm of v will also grow. Also, the norm of the updated parameter is proportional to the absoloute value of the learning rate. These observations explain the graph.

Université de Montréal
Email address: `joseph@viviano.ca`