Mathematical Foundations of Deep Neural Networks, M1407.001200
E. Ryu
Fall 2021

Homework 7
Due 5pm, Friday, November 19, 2021

Problem 1: *Transpose of downsampling.* Consider the downsampling operator $\mathcal{T}\colon \mathbb{R}^{m\times n} \to \mathbb{R}^{(m/2)\times(n/2)}$, defined as the average pool with a 2×2 kernel and stride 2. For the sake of simplicity, assume m and n are even. Describe the action of $\mathcal{T}^\top$. More specifically, describe how to compute $\mathcal{T}^\top(Y)$ for any $Y \in \mathbb{R}^{(m/2)\times(n/2)}$.

Clarification. The downsampling operator $\mathcal{T}$ is a linear operator (why?). Therefore, $\mathcal{T}$ has a matrix representation $A \in \mathbb{R}^{(mn/4)\times(mn)}$ such that

$$\mathcal{T}(X) = (A(X.\text{reshape}(mn))).\text{reshape}(m/2, n/2)$$

for all $X \in \mathbb{R}^{m\times n}$. The adjoint $\mathcal{T}^\top$ has two equivalent definitions. One definition is

$$\mathcal{T}^\top(Y) = (A^\top(Y.\text{reshape}(mn/4))).\text{reshape}(m, n)$$

for all $Y \in \mathbb{R}^{(m/2)\times(n/2)}$. Another is

$$\sum_{i=1}^{m/2}\sum_{j=1}^{n/2} Y_{ij}(\mathcal{T}(X))_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n}(\mathcal{T}^\top(Y))_{ij}(X)_{ij}$$

for all $X \in \mathbb{R}^{m\times n}$ and $Y \in \mathbb{R}^{(m/2)\times(n/2)}$.
Hint. To spoil the suspence, $\mathcal{T}^\top$ is a constant times the nearest neighbor upsampling. Explain why in your answer.

Problem 2: *Nearest neighbor upsampling.* How is the nearest neighbor upsampling operator an instance of transpose convolution? Specifically, describe how

```
layer = nn.Upsample(scale_factor=r, mode='nearest')
```

can be equivalently represented by

```
layer = nn.ConvTranspose2d(...)
layer.weight.data = ...
```

with `...` appropriately filled in.

Problem 3: *The true softmax.* Define

$$\nu_\beta(x) = \frac{1}{\beta} \log \sum_{i=1}^{n} \exp(\beta x_i).$$

Clearly, $\nu_\beta \colon \mathbb{R}^n \to \mathbb{R}$ is differentiable. Show that

(a) $\nu_\beta(x) \to \max\{x_1, \ldots, x_n\}$ as $\beta \to \infty$.

(b) $\nabla \nu_1 = \mu$, where μ is the softmax function.

(c) If $i_{\max} = \operatorname{argmax}_{1 \le i \le n} x_i$ is uniquely defined, then $\nabla \nu_\beta(x) \to e_{i_{\max}}$ as $\beta \to \infty$, where $\{e_1, \ldots, e_n\}$ is the standard basis of $\mathbb{R}^n$.

Remark. The parameter β is referred to as *inverse temperature*, since $\sum_{i=1}^{n} \exp(\beta x_i)$ is the "partition function" of statistical physics when $\beta = \frac{1}{k_B T}$, k_B is the Boltzmann constant, and T is the temperature. The regime $\beta \to \infty$ is therefore referred to as the *low-temperature regime.*

Problem 4: *Generalized inverse transform sampling.* Let $F \colon \mathbb{R} \to [0, 1]$ be the CDF of a random variable, and let $U \sim \text{Uniform}([0, 1])$. If F is strictly increasing and therefore invertible, then $F^{-1}(U)$ is a random variable with CDF F, because

$$\mathbb{P}(F^{-1}(U) \le t) = \mathbb{P}(U \le F(t))) = F(t).$$

When F is not necessarily invertible, the *generalized inverse* of F is $G \colon (0, 1) \to \mathbb{R}$ with

$$G(u) = \inf\{x \in \mathbb{R} \,|\, u \le F(x)\}.$$

Show $G(U)$ with is a random variable with CDF F.

Hint. Use the fact that F is right-continuous, i.e., $\lim_{h \to 0^+} F(x + h) = F(x)$ for all $x \in \mathbb{R}$, and that $\lim_{x \to -\infty} F(x) = 0$.

Problem 5: *f-divergence.* Let X and Y be two continuous random variables with densities p_X and p_Y. The f-divergence of X from Y is defined as

$$D_f(X \| Y) = \int f\left(\frac{p_X(x)}{p_Y(x)}\right) p_Y(x)\, dx,$$

where f is a convex function such that $f(1) = 0$.

(a) Show that $D_f(X \| Y) \ge 0$.

(b) Show that $f = -\log t$ and $f = t \log t$ correspond to KL divergence.

Problem 6: *Backprop with convolutions.* Consider 1D convolutions with single input and output channels, stride 1, and padding 0. Let $w_1, \dots, w_L$ be convolutional filters with sizes $f_1, \dots, f_L$. Let $A_{w_\ell} \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$, where $n_\ell = n_{\ell-1} - f_\ell + 1$, be the matrix representing convolution with w_ℓ, i.e., multiplication by A_{w_ℓ} is equivalent to convolution with w_ℓ, for $\ell = 1, \dots, L$. Let $\sigma \colon \mathbb{R} \to \mathbb{R}$ be a differentiable activation function. Consider the convolutional neural network

$$\begin{aligned} y_L &= A_{w_L} y_{L-1} + b_L \mathbf{1}_{n_L} \\ y_{L-1} &= \sigma(A_{w_{L-1}} y_{L-2} + b_{L-1} \mathbf{1}_{n_{L-1}}) \\ &\vdots \\ y_2 &= \sigma(A_{w_2} y_1 + b_2 \mathbf{1}_{n_2}) \\ y_1 &= \sigma(A_{w_1} x + b_1 \mathbf{1}_{n_1}), \end{aligned}$$

where $x \in \mathbb{R}^{n_0}$, $b_\ell \in \mathbb{R}$, $\mathbf{1}_{n_\ell} \in \mathbb{R}^{n_\ell}$ is the vector with all entries being 1, and $n_L = 1$. For notational convenience, define $y_0 = x$.

(a) Define

$$v_L = 1, \qquad v_\ell = \frac{\partial y_L}{\partial y_\ell} \operatorname{diag}\left(\sigma'(A_{w_\ell} y_{\ell-1} + b_\ell \mathbf{1}_{n_\ell}\right) \qquad \text{for } \ell = 1, \dots, L-1.$$

Let $\mathcal{C}_{v_\ell^\intercal}$ be the 1D convolutional operator defined by interpreting $v_\ell^\intercal \in \mathbb{R}^{n_\ell}$ as a convolutional filter for $\ell = 1, \dots, L$. Show that

$$\begin{aligned} &\frac{\partial y_L}{\partial y_{L-1}} = A_{w_L}, \qquad \frac{\partial y_\ell}{\partial y_{\ell-1}} = \operatorname{diag}\left(\sigma'(A_{w_\ell} y_{\ell-1} + b_\ell \mathbf{1}_{n_\ell})\right) A_{w_\ell} \qquad \text{for } \ell = 2, \dots, L-1 \\ &\frac{\partial y_L}{\partial w_\ell} = (\mathcal{C}_{v_\ell^\intercal} y_{\ell-1})^\intercal \qquad \text{for } \ell = 1, \dots, L \\ &\frac{\partial y_L}{\partial b_\ell} = v_\ell \mathbf{1}_{n_\ell} \qquad \text{for } \ell = 1, \dots, L. \end{aligned}$$

(b) As discussed in homework 1, forming the full matrix A_{w_ℓ} is wasteful and should be avoided. Describe how matrix-vector or vector-matrix products with respect to A_{w_i} or $A_{w_i}^\intercal$ should be used in the forward pass and backpropagation.

Clarification. A matrix-vector product $A_{w_i} v$ should be computed by performing convolution. A vector-matrix product $u^\intercal A_{w_i} = (A_{w_i}^\intercal u)^\intercal$ should be computed by performing transpose-convolution, which was discussed in homework 1.

Figure 1: KMNIST images

Problem 7: *Anomaly detection via AE.* In this problem, you will use an autoencoder to perform anomaly detection between the MNIST and the Kuzushiji(崩し字)-MNIST (KMNIST) [1] datasets. KMNIST contains handwritten Japanese characters. Download the starter code `anomaly_detection.py` and implement the following steps. In step 1, load the MNIST and KMNIST dataset, and split the MNIST training dataset into "training" and "validation" set. (Together with the "test" set you will have three datasets in total.) In step 2, define the AE model. In step 3, instantiate the model and select the Adam optimizer. In step 4, train the AE with the training data $X_1, \ldots, X_N$ with loss

$$\ell(\theta, \varphi) = \sum_{i=1}^{N} \|X_i - D_\varphi(E_\theta(X_i))\|^2,$$

where E_θ is the encoder and D_φ is the decoder. Do not use the validation set in this stage. In step 5, define the score function

$$s(X) = \|X - D_\varphi(E_\theta(X))\|^2$$

and calculate the mean and standard deviation of

$$\{s(Y_i)\}_{i=1}^{M}$$

where $Y_1, \ldots, Y_M$ are the validation data. Define a threshold to be mean + 3 standard deviations, and define inputs with score function value exceeding this threshold to be anomalies. In step 6, check how many of the MNIST images within the test set are classified as anomalies and report the type I error rate. In step 7, check how many of the KMNIST images are classified as anomalies and report the type II error rate.

References

[1] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha, Deep learning for classical Japanese literature, *NeurIPS ML for Creativity Workshop*, 2018.