# TTIC 31230 Fundamentals of Deep Learning, 2020
## Problems For Trainability.

In these problems, as in the lecture notes, capital letter indeces are used to indicate subtensors (slices) so that, for example, $M[I, J]$ denotes a matrix while $M[i, j]$ denotes one element of the matrix, $M[i, J]$ denotes the $i$th row, and $M[I, j]$ denotes the $j$th collumn.

Throughout these problems we assume a word embedding matrix $e[W, I]$ where $e[w, I]$ is the word vector for word $w$. We then have that $e[w, I]^\top h[t, I]$ is the inner product of the word vector $w[w, I]$ and the hidden state vector $h[t, I]$.

We will adopt the convention, similar to true Einstein notation, that repeated capital indeces in a product of tensors are implicitly summed. We can then write the inner product $e[w, I]^\top h[t, I]$ simply as $e[w, I]h[t, I]$ without the need for the (meaningless) transpose operation.

**Problem 1.** This problem is on initialization. Consider a single unit defined by

$$u = f\left(\left(\sum_{i=1}^{N} W[i]x[i]\right) - B\right).$$

where $B$ is initialized to zero and $f$ is an activation function such as a sigmoid or ReLU. The vector $x$ is a random variable determined by a random draw of a training example. Assume that the components of $x$ are independent and that each component has zero mean and unit variance. Suppose that we initialize each weight in $W$ from a distribution with zero mean and variance $\sigma^2$ and that the distribution is symmetric about zero — (the probability that $w[i] = z$ equals the probability that $w[i] = -z$). For example, $x[i]$ might be distributed as a zero-mean unit-variance Gaussian. Consider $y = \sum_i W[i]x[i]$ as a random variable defined by the distribution on $x$ and the independent random distribution on $W$. Recall that the variance $\sigma^2$ of a sum of independent random variables is the sum of the variances and the variance of a product of zero mean independent random variables is the product of the variances.

(a) What value of $\sigma$ for $W[i]$ gives zero mean and unit variance for $y$ if the vectors $w[I]$ and $x[I]$ have dimension $d$? Show your derivation.

(b) For a sigmoid activation function what is the mean of $u$.

(c) For a sigmoid activation function is the variance of $u$ larger than, equal to, or smaller than the variance of $y$?

(d) What is the largest possible variance of the output of a sigmoid?

**Problem 2.** Consider a regression problem where we want to predict a scalar value $y$ from a vector $x$. Consider the L-layer perceptron for this problem defined by the following equations which compute hidden layer vectors $h_1[I], \ldots, h_L[I]$

and predictions $\hat{y}_1, \ldots, \hat{y}_L$ where the prediction $\hat{y}_\ell$ is done with a linear regression on the hidden vector $h_\ell[I]$.

$$
\begin{aligned}
h_0[i] &= x[i] \\
&\vdots \\
h_{\ell+1}[i] &= \sigma(W^{h,h}_{\ell+1}[i, I]h_\ell[I] - B^{h,h}_{\ell+1}[i]) \\
\hat{y}_{\ell+1} &= W^{h,p}_{\ell+1}[I]h_{\ell+1}[I] - B^{h,y}_{\ell+1} \\
&\vdots \\
\text{Loss} &= \sum_{\ell=1}^{L}(y - \hat{y}_\ell)^2
\end{aligned}
$$

Each term $(y - \hat{y}_\ell)^2$ is called a "loss head" and defines a loss on each prediction $\hat{y}_\ell$. Note, however, that there is only one scalar loss minimized by SGD which is the sum of the losses of each loss head.

**(a) 7 pts** Explain why these multiple loss terms might improve the ability of SGD to find a useful $L$-layer MLP regrssion $\hat{y}_L$ when $L$ is large.

**(b) 7 pts** As a function of $L$ (ignoring the dimension size $I$) what is the order of run time for the backpropagation procedure. Explain your answer.

**(c) 11 pts** Rewrite the above MLP equations to use residual connections rather than multiple heads. There are multiple correct solutions differing in minor details. Pick one that seems good to you.

**Problem 3.** This problem is on the initialization of ResNet filters. Consider the following residual skip connection where $R_{\ell+1}$ is computed with an $N \times N$ filter.

$$
\begin{aligned}
&\text{for } b, x, y, j\Delta x, \Delta y, j' \\
&\qquad R_{\ell+1}[b, x, y, j] \quad \mathrel{+}= \quad W_{\ell+1}[\Delta x, \Delta y, j', j] \; L_\ell[b, x + \Delta x, y + \Delta y, j'] \\[1em]
&\quad \text{for } b, x, y, j \\
&\qquad R_{\ell+1}[b, x, y, j] \quad \mathrel{-}= \quad B_{\ell+1}[j] \\[1em]
&\quad \text{for } b, x, y, j \\
&\qquad L_{\ell+1}[b, x, y, j] \quad = \quad L_\ell[b, x, y, j] + R_{\ell+1}[b, x, y, j]
\end{aligned}
$$

Here we have omitted an activation function that would be present in practice. This omission allows an analysis that seems to provide insight into the more complex case with activations.

Assume that $L_0[b, x, y, j]$ is computed from the input in some unspecified way such that $L_0[b, x, y, j]$ has unit variance. Assume that the values $L_\ell[b, x, y, j]$ and $R_{\ell+1}[b, x, y, j]$ are all independent. Suppose that each weight $W_\ell[\Delta x, \Delta y, j, j']$ is drawn independently at random from a distribution with zero mean and variance $\sigma_W$. Recall that the variance $\sigma^2$ of a sum of independent random is the sum of the variances and the variance of a product of independent random variables is the product of the variances.

(a) Give an expression for the variance $\sigma_\ell^2$ of $L_{\ell+1}[b, x, y, j]$ as a function of $\ell$, the filter dimension $D = \Delta X = \Delta Y$, the feature dimension $J$, and the weight variance $\sigma_W^2$.

(b) Using $(1 + \epsilon)^N \approx e^{\epsilon N}$ solve for the value of $\sigma_W$ such that $\sigma_L^2 = 2$.

(c) Assuming $L_L.\mathrm{grad}[b, x, y, j]$ has unit variance, and that all components of $L_\ell.\mathrm{grad}[b, x, y, j]$ and $R_\ell.\mathrm{grad}[b, x, y, j]$ are independent, give an expression for the variance $\sigma_{\ell,\mathrm{grad}}^2$ of the components of $L_\ell.\mathrm{grad}[b, x, y, j]$ as a function of $\ell$, $D$, $J$ and $\sigma_W$.

**Problem 4. Gated CNNs**
A UGRNN is defined by the following equations.

$$\tilde{R}_t[b, j] = \left( \sum_i W^{h,R}[j, i] h_{t-1}[b, i] \right) + \left( \sum_k W^{x,R}[j, k] x_t[b, k] \right) - B^R[j]$$

$$R_t[b, j] = \tanh(\tilde{R}_t[b, j])$$

$$\tilde{G}_t[b, j] = \left( \sum_i W^{h,G}[j, i] h_{t-1}[b, i] \right) + \left( \sum_k W^{x,G}[j, k] x_t[b, k] \right) - B^G[j]$$

$$G_t[b, j] = \sigma(\tilde{G}_t[b, j])$$

$$h_t[b, j] = G_t[b, j] h_{t-1}[b, j] + (1 - G_t[b, j]) R_t[b, j]$$

Modify these to form a data-dependent data-flow CNN for vision — an Update-Gate CNN (UGCNN). More specifically, give equations analogous to those for UGRNN for computing a CNN "box" $L_{\ell+1}[b, x, y, j]$ from $L_\ell[b, x, y, i]$ (stride 1) using a computed "gate box" $G_{\ell+1}[b, x, y, j]$ and an "update box" $R_{\ell+1}[b, x, y, j]$.

$$\Phi = (W_{\ell+1}^{L,R}[\Delta x, \Delta y, j, j'], B_{\ell+1}^R[j], W_{\ell+1}^{L,G}[\Delta x, \Delta y, j, j'], B_{\ell+1}^G[j])$$

**Problem 5.** RNNs
Below are the equations defining the update cell of the UGRNN which takes as data inputs $h[B, t-1, I]$ and $x[B, t, K]$ and produces $h[B, t, J]$.

$$R[b,t,j] \;=\; \tanh\left(W^{h,R}[j,I]h[b,t{-}1,I] + W^{x,R}[j,K]x[b,t,K] - B^R[j]\right)$$

$$G^h[b,t,j] \;=\; \sigma\left(W^{h,G}[j,I]h[b,t{-}1,I] + W^{x,G}[j,K]x[b,t,K] - B^G[j]\right)$$

$$h[b,t,j] \;=\; G^h[b,t,j]h[b,t{-}1,j] + (1 - G^h[b,t,j])R[b,t,j]$$

Here I have written the gate as $G^h$ to emphasize that it is a gate used to define a convex combination of the $h_{t-1}$ and $x_t$ inputs to $h_t$. Modify these equations to use a second gate $G^R$ which gates inputs to $R$ so that the input to the activation function (threshold function) producing $R[b,t,j]$ is a convex combination of

$$W^{h,R}[j,I]h[b,t{-}1,I] - B^{h,R}[j]$$

and

$$W^{x,R}[j,K]x[b,t,K] - B^{x,R}[j]$$

where the weighting in the convex combination is given by your new gate $G^R[b,t,j]$. This gated RNN is similar to, but different from, a GRU which also has two gates.

**Problem 6. RNN parallel run time.** Consider an autoregressive RNN neural language model with $P_\Phi(w_{t+1}|w_1,\ldots,w_t)$ defined by

$$P_\Phi(w_t|w_1,\ldots,w_{t-1}) = \operatorname*{softmax}_{w_{t+1}}\; e[w_t,I]h[t-1,I]$$

Here $e[w,I]$ is the word vector for word $w$, $h[t,I]$ is the hidden state vector at time $t$ of a left-to-right RNN, and as described above $e[w,I]h[t,I]$ is the inner prodcut of these two vectors where we have assumed that they have the same dimension. For the first word $w_1$ we have an externally provided initial hidden state $h[0,I]$ and $w_1,\ldots,w_0$ denotes the empty string. We train the model on the full loss

$$\Phi^* \;=\; \operatorname*{argmin}_{\Phi} E_{w_1,\ldots,w_T \sim \text{Train}}\; -\ln P_\Phi(w_1,\ldots,w_T)$$

$$=\; \operatorname*{argmin}_{\Phi} E_{w_1,\ldots,w_T \sim \text{Train}} \sum_{t=1}^{T} -\ln P_\Phi(w_t|w_1,\ldots,w_{t-1})$$

What is the order of run time as a function of sentence length $T$ for the back-propagation for this model run on a sentence $w_1,\ldots,w_T$? Explain your answer.
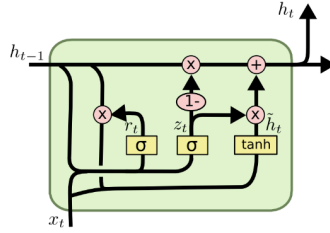
**Problem 7. Translating diagrams into equations.** A UGRNN cell for computing $h[b,t,J]$ from $h[b,t-1,J]$ and $x[b,t,J]$ can be written as

$$G[b,t,j] \;=\; \sigma\left(W^{h,G}[j,I]h[b,t{-}1,I] + W^{x,G}[j,K]x[b,t,K] - B^{G}[j]\right)$$

$$R[b,t,j] \;=\; \tanh\left(W^{h,R}[j,I]h[b,t{-}1,I] + W^{x,R}[j,K]x[b,t,K] - B^{R}[j]\right)$$

$$h[b,t,j] \;=\; G[b,t,j]h[b,t{-}1,j] + (1 - G[b,t,j])R[b,t,j]$$

Modify the above equations so that they correspond to the following diagram for a Gated Recurent Unit (GRU).



[Christopher Olah]