Practical Lecture 7 - Convolution Networks, Recurrent Networks

Luis Sa-Couto[1] and Andreas Wichert[2]

INESC-ID, Instituto Superior Tecnico, Universidade de Lisboa
{luis.sa.couto,andreas.wichert}@tecnico.ulisboa.pt

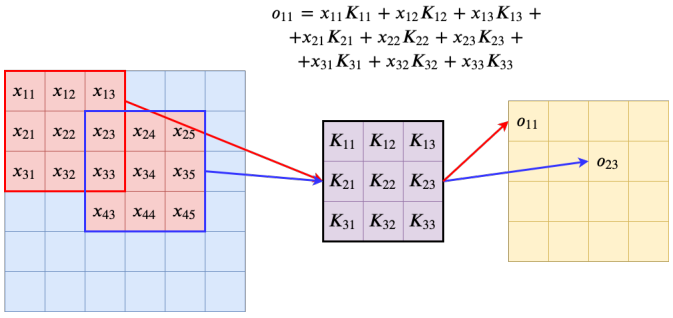
1 Convolution Networks

Convolution Networks are a way to structure a network's architecture such that it uses knowledge from how images are built to reduce the number of free parameters and thus increase generalization. The cornerstone of these networks is the convolution operation. Let us recall the convolution operation.

Convolution Operation: The convolution which is represented by the symbol $*$. Given a $k_h \times k_w$ kernel matrix $\mathbf{K}_{ij}$. For a given input matrix $\mathbf{x} \in \mathbb{R}^{H \times W}$, the convolution of $\mathbf{x}$ by $\mathbf{K}$ is as a matrix $\mathbf{o} = \mathbf{x} * \mathbf{K}$ where the coordinates are defined as:

$$o_{ij} = (\mathbf{x} * \mathbf{K})_{ij} = \sum_{h=1}^{k_h} \sum_{w=1}^{k_w} x_{(i+h-1)(j+w-1)} K_{hw}$$

The following figure presents an example where we show the computation performed by convolution to get two output positions.

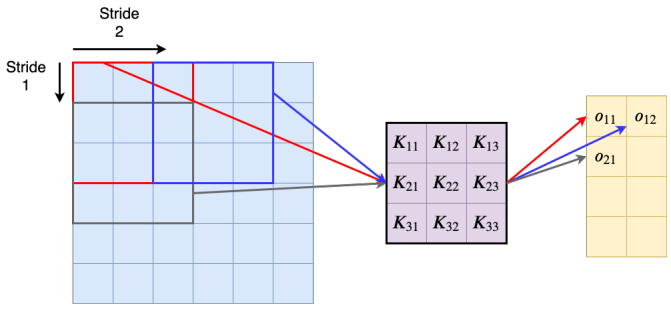


We can think of each step as a dot product between the kernel and the image's window. So, the result will be high if the window is similar to the kernel. This tells us that convolving an image with a kernel corresponds to searching for occurrences of the feature represented by that kernel in the image.

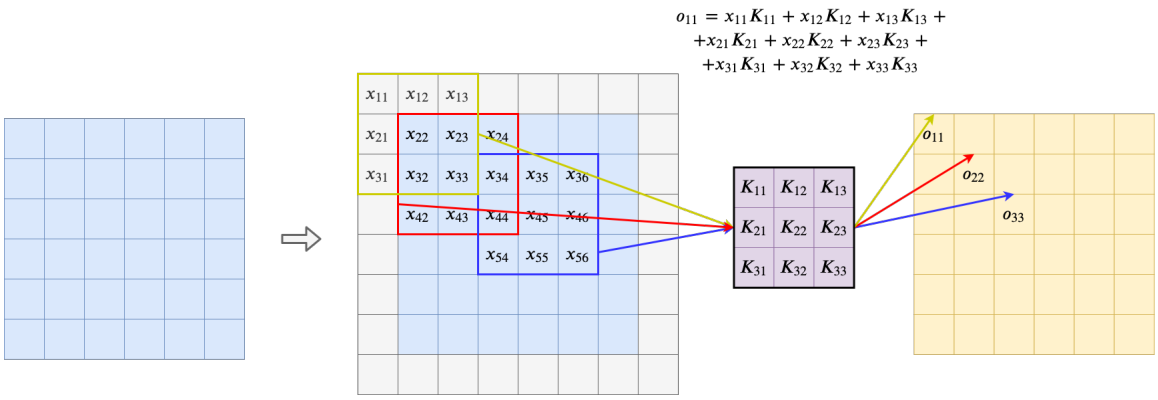
Looking at the way convolution is defined, we can see that it corresponds to sliding a window with the same size as the kernel through the image. Concretely,

the window start at the top left corner and its contents are multiplied by the kernel and summed. This process achieves the top left corner of the output. Afterwards, the window shifts one position to right and the same process is applied. Eventually, we can no longer slide the window to the right without crossing the input boundary. Then, the window goes back to the left side but shited down by one position. If we define the amount that the window is shifted accross the image's width from step to step s_w, we say that the regular convolution has a horizontal stride of one $s_w = 1$. The same reasoning tell us that the vertical (or height) stride $s_h = 1$ as well.

Variable Stride: With the previous definition in mind, we can change the stride value and move the window more from step to step. This may cause convolution to skip input positions. The following figure provides an example where $s_w = 2$ and $s_h = 1$.

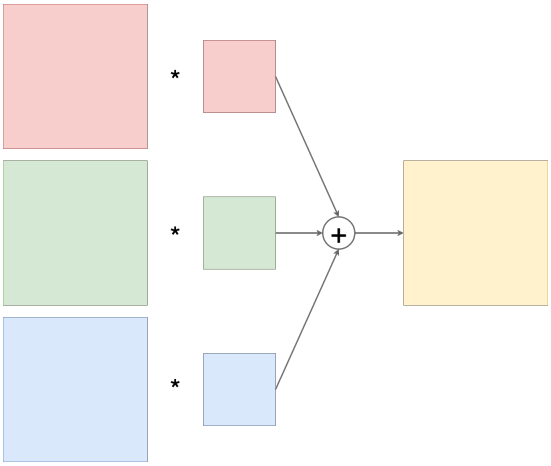


Padding: If we want convolution to look around all possible input positions, we can use a technique called padding. The trick is to add a frame of invented pixels around the image. Several values can be placed on these pixels. A typical choice is to put all zeros which is called zero padding. The following figure shows convolution applied to a padded image where $p_h = 1$ because we added a row of imaginary pixels above and below. Also, we have that $p_w = 1$ because we added a column on the left and on the right.

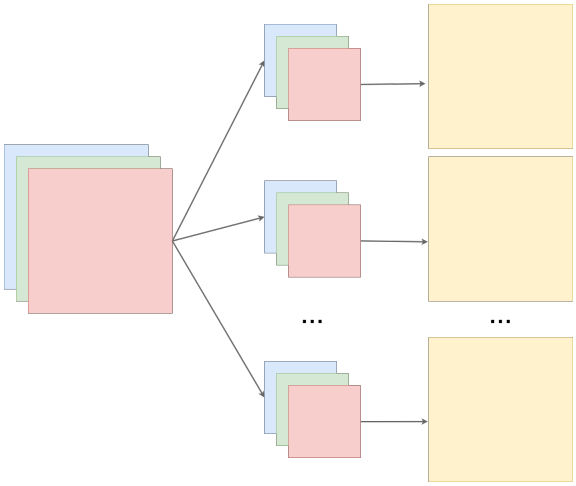


Convolution with Channels: Usually, convolution is not applied to a single two-dimensional matrix but to a collection of matrices where each is called a channel. Think about an image in RGB where each color has its matrix of pixels. In this case, each color is a channel.

In this formulation, the kernel has the same number of channels as the input. Now, convolution between an input with channels and a kernel with channels corresponds to applying the old convolution between the corresponding channels independently and summing the results. The figure below provides an example.

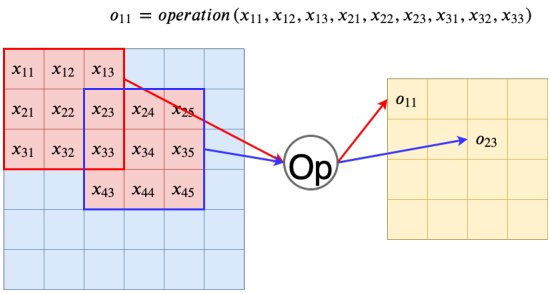


Convolution Layer: A convolution layer computes the convolution operation between an input and a set of different kernels. Since each convolution results in a two-dimensional matrix (one for each kernel), we say that the output of a layer with k kernels is an new representation of an image with k channels.



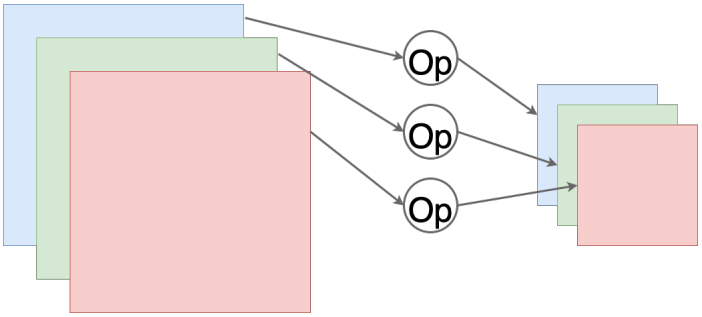
Thinking about what the layer is doing, we see that each channels correspond to a map that pinpoints where the feature represented by that kernel seems to appear.

Pooling Operation: The pooling operation is very similar to the convolution operation. However, there is no kernel. For each window in the input, a fixed operation is applied. Usually, two kinds of pooling are used: max pooling and average pooling. For the figure below, the first one would correspond to $o_{11} = \max(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33})$. Where as the second one would be $o_{11} = mean(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33})$.



This process is usually done to change the resolution. This not only allows for noise to fade but also yields a smaller image, which will result in less parameters and thus more generalization. However, if the pooling operation is over done, the resolution is too small and we may lose not only noise but also very important features.

Pooling Layer: A pooling layer applies the pooling operation to every channel independently. The figure below provides an illustration.



1) a) For a kernel of shape $(k_h, k_w) = (5, 3)$ with one channel, a stride of $(s_h, s_w) = (1, 1)$ and an input of shape $(H, W, C) = (6, 5)$ with one channel, what padding (p_h, p_w) is required such that the output has the same shape as the input (also called same padding)?

Solution:

The input will be something like the following:

$$\begin{pmatrix} I_{110} & I_{120} & I_{130} & I_{140} & I_{150} \\ I_{210} & I_{220} & I_{230} & I_{240} & I_{250} \\ I_{310} & I_{320} & I_{330} & I_{340} & I_{350} \\ I_{410} & I_{420} & I_{430} & I_{440} & I_{450} \\ I_{510} & I_{520} & I_{530} & I_{540} & I_{550} \\ I_{610} & I_{620} & I_{630} & I_{640} & I_{650} \end{pmatrix}$$

where I_{hwc} corresponds to the input value at height h, width w and channel c.

Let us focus on one dimension at a time.

For the height, a general slice at width w and channel c will have the following values:

$$\begin{pmatrix} I_{1wc} \\ I_{2wc} \\ I_{3wc} \\ I_{4wc} \\ I_{5wc} \\ I_{6wc} \end{pmatrix}$$

If we center the kernel at all possible heights we get the following possibilities:

$$\begin{pmatrix} ? \\ ? \\ I_{1wc} \\ I_{2wc} \\ \underline{I_{3wc}} \\ I_{4wc} \\ I_{5wc} \\ I_{6wc} \end{pmatrix}, \begin{pmatrix} ? \\ I_{1wc} \\ I_{2wc} \\ I_{3wc} \\ \underline{I_{4wc}} \\ I_{5wc} \\ I_{6wc} \end{pmatrix}, \begin{pmatrix} I_{1wc} \\ I_{2wc} \\ I_{3wc} \\ I_{4wc} \\ \underline{I_{5wc}} \\ I_{6wc} \end{pmatrix}, \begin{pmatrix} I_{1wc} \\ \underline{I_{2wc}} \\ I_{3wc} \\ I_{4wc} \\ I_{5wc} \\ \underline{I_{6wc}} \end{pmatrix}, \begin{pmatrix} I_{1wc} \\ \underline{I_{2wc}} \\ I_{3wc} \\ I_{4wc} \\ I_{5wc} \\ I_{6wc} \\ ? \end{pmatrix}, \begin{pmatrix} I_{1wc} \\ I_{2wc} \\ \underline{I_{3wc}} \\ I_{4wc} \\ I_{5wc} \\ I_{6wc} \\ ? \\ ? \end{pmatrix}$$

So, we will need two extra dimensions on both sides to cover the top and bottom cases. For that reason, $p_h = 2$.

Repeating the same reasoning for the width and the channels, we get $p_w = 1$ and $p_c = 0$.

---

b) For a general kernel shape (k_h, k_w), a general input shape (H, W) and a fixed stride of $(s_h, s_w) = (1, 1)$. Can you derive a general formula for (p_h, p_w) such that the output has shape (H, W)?

Solution:

For an input of height H, if we want the output to have also height H we will have:

$$H = H - f_h + 2p_h + 1 \iff p_h = \frac{f_h - 1}{2}$$

For the width we will have:

$$W = W - f_w + 2p_w + 1 \iff p_w = \frac{f_w - 1}{2}$$

---

2) Connsider the following network:

- Input:
  - $(H, W, C) = (28, 28, 3)$
- Convolution:
  - Number of kernels: 8
  - For all kernels the shapes are: $(k_h, k_w, k_c) = (5, 5, 3)$
  - The strides are: $(s_h, s_w) = (1, 1)$
  - Same padding
- Pooling:
  - $(k_h, k_w) = (4, 4)$
  - $(s_h, s_w) = (2, 2)$

a) List and count all parameteres in this network.

Solution:

Each kernel channels is a matrix of shape 5×5. Each kernel has three channels. So, each kernel will require $3 \times 5 \times 5 = 75$ parameters.

The convolution layer has 8 kernels, so the total amount of parameters for all kernels is $8 \times 75 = 600$.

Besides the kernel weights, there will also be biases. More specifically, 8biases, one per kernel. So, $600 + 8 = 608$ parameters.

The pooling layer has a fixed operation, so there are no parameters.

In total, the network has 608 parameters.

---

b) What is the output's shape?

Solution:

Since the convolution layer has same padding, the output of that layer will have the same height and width of the input 28×28. The number of kernels in the layer is 8 so the output of the layer will be 8 channels of shape 28×28.

The pooling layer applies a pooling operation to each channel independently. So, the number of channels will not change. However, the height and width do change.

Horizontally, we can move a 4×4 window with a two pixel stride for 12 positions before we cross the input boundary. Namely, we will have windows starting at widths $1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23$.

Vertically we get the same thing. So, the final output of the whole network corresponds to 8channels of shape 12×12.

---

3) Connsider the following network:

- Input:
  - $(H, W, C) = (10, 10, 1)$
- Convolution:
  - Number of channels: 8
  - $(k_h, k_w, k_c) = (5, 5, 1)$
  - $(s_h, s_w) = (1, 1)$
  - Same padding
- Convolution:
  - Number of channels: 12
  - $(k_h, k_w, k_c) = (3, 3, 8)$
  - $(s_h, s_w) = (1, 1)$
  - Same padding

a) List and count all parameteres in this network.

Solution:

The first convolution layer has 8 kernels with one channel with shape $(5, 5)$. So, there will be eight matrices of shape 5×5. So, a total of $8 \times 5 \times 5 = 200$ parameters.

Besides the weights, there will also be biases. More specifically, 8biases, one per kernel. So, a total of $200 + 8 = 208$ parameters.

The second convolution layer has 12 kernels where each has 8 channels with shape $(3, 3)$ and 12 biases. So, the second layer will have $12 \times 8 \times 3 \times 3 + 12 = 876$ parameters.

Finally, the whole network will have a total of $208 + 876 = 1084$ parameters.

---

b) What is the output's shape?

Solution:

Since the first convolution layer has same padding the heigth and width do not change. So, with 8 kernels, the output will have 8 channels where each has a shape of 10×10.

The second convolution layer is similar. The same padding will keep the height and width constant. With 12 channels, the final output will have 12 channels of 10×10 values.

---

4) Given an input image of shape $(10, 10, 1)$ connected to a layer of $(10, 10, 5)$ units.

a) How many parameters does the network have if it is fully connected?

Solution:

Every input unit will be connected to all units in the next layer. So, there will be $(10 \times 10 \times 1) \times (10 \times 10 \times 5)$ weight connections. Also, each unit in the layer has its bias term, so there will be $10 \times 10 \times 5$ bias parameters.

---

b) What about if it is a convolution layer with 5 kernels of shape $(3, 3, 1)$ with same padding and unitary stride.

Solution:

The convolution layer has 5 kernels where each has shape $(3, 3, 1)$ and 3 biases. So, the total number of parameters is $5 \times 1 \times 3 \times 3 + 3 = 48$.

---

5) For the following input image:

$$I = \begin{pmatrix} 10 & 10 & 10 & 10 & 10 & 10 \\ 5 & 5 & 5 & 5 & 5 & 5 \\ 2 & 2 & 2 & 0 & 0 & 2 \\ 2 & 2 & 2 & 0 & 0 & 2 \\ 50 & 10 & 50 & 50 & 50 & 10 \\ 50 & 10 & 50 & 50 & 50 & 10 \end{pmatrix}$$

a) What is the output provided by a convolution layer with the following properties:

- Zero Padding: $(1, 1)$
- Stride: $(1, 1)$
- Kernel Shape: $(3, 3)$
- Number of kernels: 2
- Kernels:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Solution:
First, let us compute the output dimensions:

$$Output_h = \left\lfloor \frac{6 - 3 + 2 \times 1}{1} \right\rfloor + 1 = 6$$

$$Output_w = \left\lfloor \frac{6 - 3 + 2 \times 1}{1} \right\rfloor + 1 = 6$$

The first channel of the output will be:

$$pad\,(I, (1, 1)) * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 10 & 10 & 10 & 10 & 10 & 0 \\ 0 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \\ 0 & 2 & 2 & 2 & 0 & 0 & 2 & 0 \\ 0 & 2 & 2 & 2 & 0 & 0 & 2 & 0 \\ 0 & 50 & 10 & 50 & 50 & 50 & 10 & 0 \\ 0 & 50 & 10 & 50 & 50 & 50 & 10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 20 & 35 & 35 & 35 & 35 & 20 \\ 29 & 46 & 44 & 42 & 42 & 27 \\ 16 & 25 & 21 & 19 & 19 & 12 \\ 66 & 120 & 116 & 154 & 114 & 62 \\ 74 & 216 & 174 & 252 & 172 & 112 \\ 70 & 210 & 170 & 250 & 170 & 110 \end{pmatrix}$$

The second channel of the output will be:

$$pad\,(I,(1,1)) * \begin{pmatrix} 0&0&0\\0&1&0\\0&0&0 \end{pmatrix} = \begin{pmatrix} 0&0&0&0&0&0&0&0\\0&10&10&10&10&10&10&0\\0&5&5&5&5&5&5&0\\0&2&2&2&0&0&2&0\\0&2&2&2&0&0&2&0\\0&50&10&50&50&50&10&0\\0&50&10&50&50&50&10&0\\0&0&0&0&0&0&0&0 \end{pmatrix} * \begin{pmatrix} 0&0&0\\0&1&0\\0&0&0 \end{pmatrix} = \begin{pmatrix} 10&10&10&10&10&10\\5&5&5&5&5&5\\2&2&2&0&0&2\\2&2&2&0&0&2\\50&10&50&50&50&10\\50&10&50&50&50&10 \end{pmatrix}$$

---

b) Take the output from **a)** and apply a max pooling layer with the following properties:

- Zero Padding: $(0,0)$
- Stride: $(2,2)$
- Kernel Shape: $(2,2)$

Solution:
First, let us compute the output dimensions:

$$Output_h = \left\lfloor \frac{6-2+2\times 0}{2} \right\rfloor + 1 = 3$$

$$Output_w = \left\lfloor \frac{6-2+2\times 0}{2} \right\rfloor + 1 = 3$$

The first channel of the output will be:

$$maxpool \begin{pmatrix} 20&35&35&35&35&20\\29&46&44&42&42&27\\16&25&21&19&19&12\\66&120&116&154&114&62\\74&216&174&252&172&112\\70&210&170&250&170&110 \end{pmatrix} = \begin{pmatrix} 46&44&42\\120&154&114\\216&252&172 \end{pmatrix}$$

The second channel of the output will be:

$$maxpool \begin{pmatrix} 10&10&10&10&10&10\\5&5&5&5&5&5\\2&2&2&0&0&2\\2&2&2&0&0&2\\50&10&50&50&50&10\\50&10&50&50&50&10 \end{pmatrix} = \begin{pmatrix} 10&10&10\\2&2&2\\50&50&50 \end{pmatrix}$$

---

6) For the following kernels, describe what kind of feature they extract from the image:

$$F_1 = \begin{pmatrix} -10 & -10 & -10 \\ 5 & 5 & 5 \\ -50 & -10 & -10 \end{pmatrix}, F_2 = \begin{pmatrix} 2 & 2 & 2 \\ 2 & -12 & 2 \\ 2 & 2 & 2 \end{pmatrix}, F_3 = \begin{pmatrix} -20 & -10 & 0 & 5 & 10 \\ -10 & 0 & 5 & 10 & 5 \\ 0 & 5 & 10 & 5 & 0 \\ 5 & 10 & 5 & 0 & -10 \\ 10 & 5 & 0 & -10 & -20 \end{pmatrix}$$

Solution:

F_1: Thin horizontal line

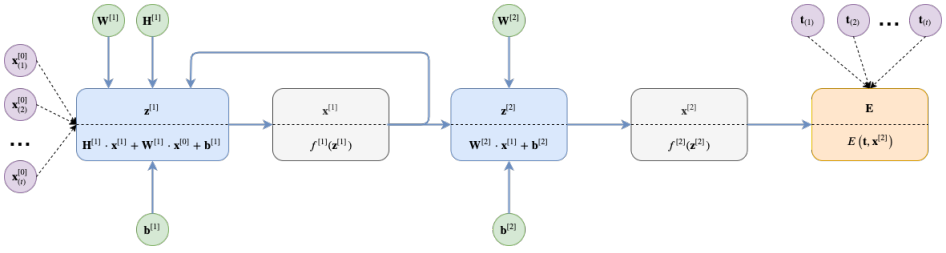
F_1: Frame with no center

F_1: Diagonal line

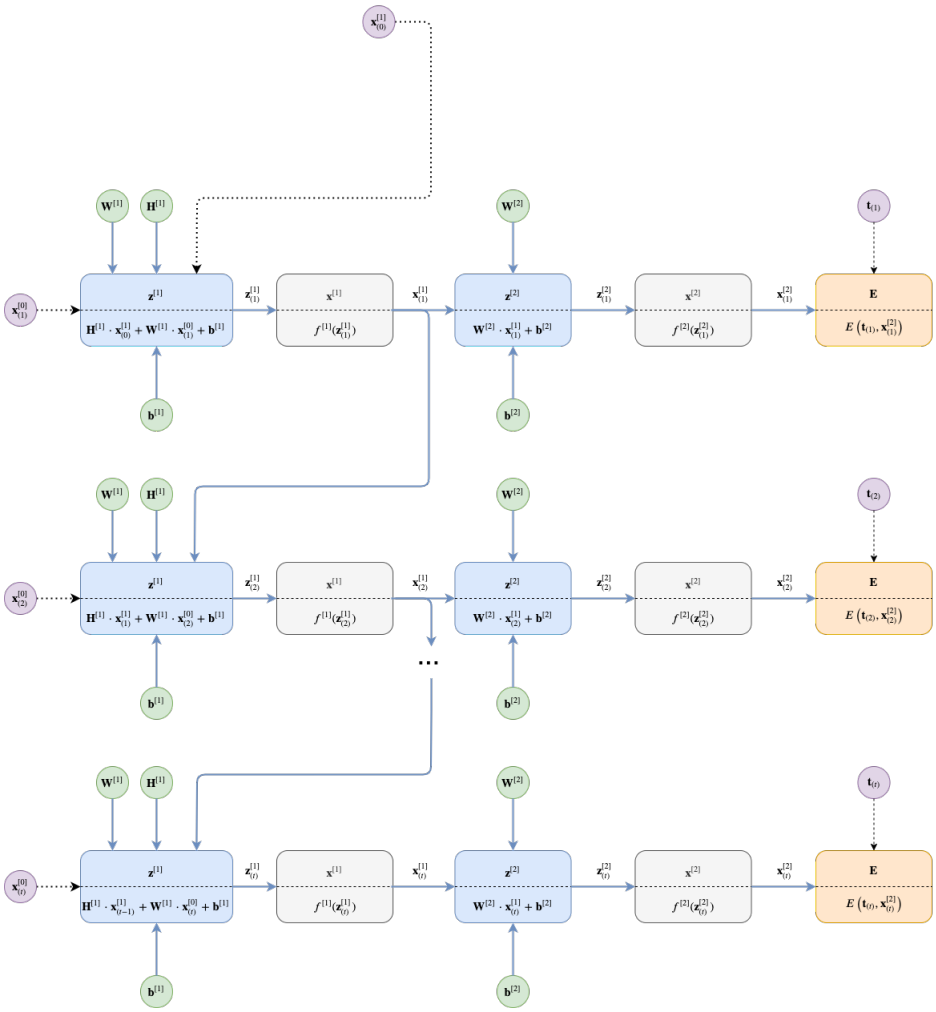
---

2 Recurrent Networks

Below we present a simple recurrent network that maps a sequence of inputs to a sequence of targets. In this example we use only one hidden layer that has a feedback connection.



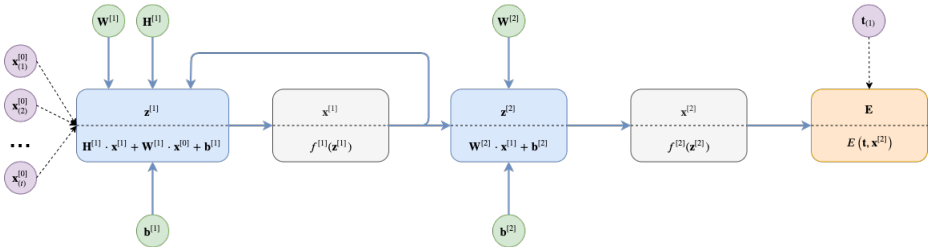
When one thinks about how to apply backpropagation to learn the parameters on such a network it may not seem obvious at first. However, if we unfold the network's function graph, the recurrent network becomes a much like a regular one.



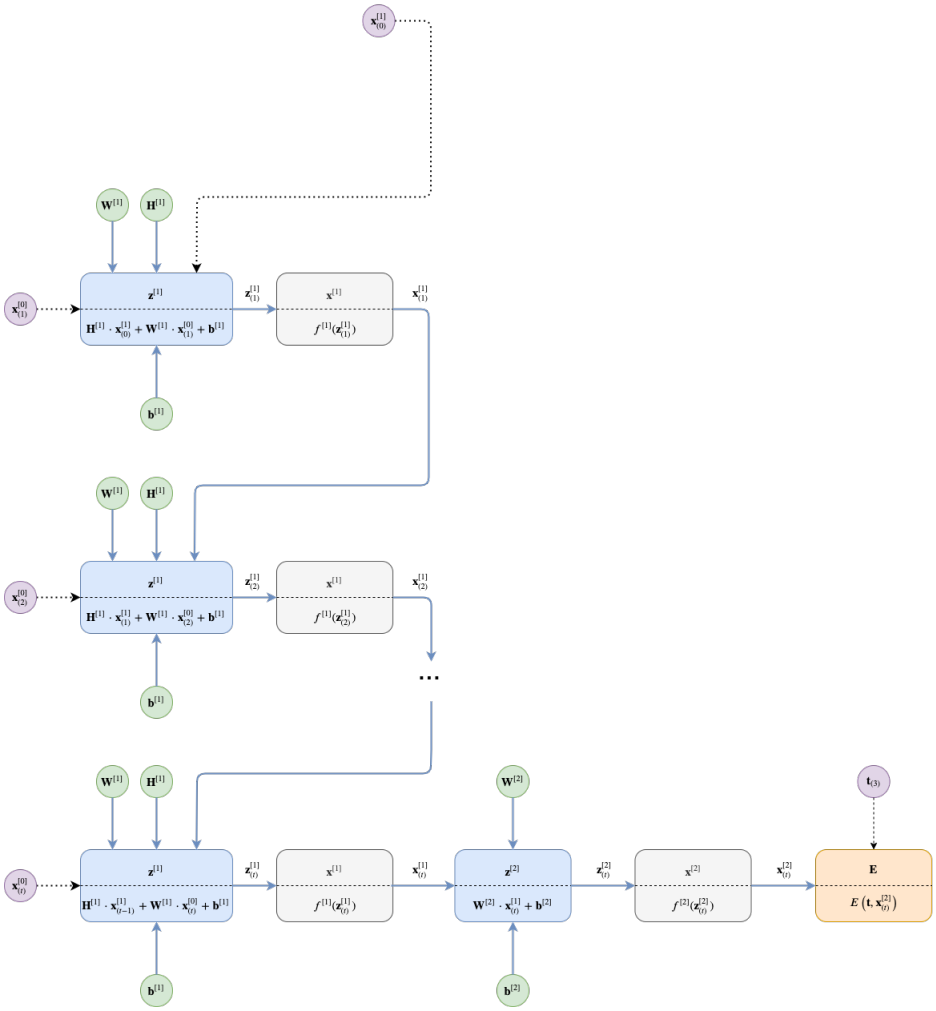
Unfolding corresponds to making copies of the network structure to match the total number of sequence elements. This implies that between copies of the network the weights are shared! This means that $\mathbf{W}^{[1]}$ at the bottom of the unfolded graph is the same as the $\mathbf{W}^{[1]}$ at the top. This weight sharing will influence the way we apply the backward pass. Namely, to update a parameter, we need to sum accross all paths that lead to all copies of that parameter. A requirment of this approach is that we need to initialize $\mathbf{x}_{(0)}^{[1]}$ to feed as hidden state to the first copy of the network.

As an example, we used a many-to-many recurrent network and showed unfolding corresponds to making copies of its structure for every timestep. However, many other architecture templates exist. For example, we could have a many-to-

one network. In this case, we will see that sometimes what we have to replicate is just the recursion part. For example, for the network:



the unfolding would be done as follows:



Note that the recursion is copied throughout all timesteps but since there is only one output, we only place an output layer after the last timestep.

On the exercises below we will go through the details of both of these architectures.

1) Consider a recurrent network with one hidden layer to solve a **many-to-many** task. Take into account that:

- The recurrent connections happen between the hidden layers
- The hidden activation function is the hyperbolic tangent
- The output activation function is softmax
- The error function is the cross-entropy between output and target

a) Write down the model equations for forward propagation.

Solution:

For all $t \in \{1, \dots, T\}$:

$$\begin{aligned}
\mathbf{z}^{[1]}_{(t)} &= z^{[1]}\left(\mathbf{H}^{[1]}, \mathbf{x}^{[1]}_{(t-1)}, \mathbf{W}^{[1]}, \mathbf{x}^{[0]}_{(t)}, \mathbf{b}^{[1]}\right) &&= \mathbf{H}^{[1]}\mathbf{x}^{[1]}_{(t-1)} + \mathbf{W}^{[1]}\mathbf{x}^{[0]}_{(t)} + \mathbf{b}^{[1]} \\
\mathbf{x}^{[1]}_{(t)} &= f^{[1]}\left(\mathbf{z}^{[1]}_{(t)}\right) &&= \tanh\left(\mathbf{z}^{[1]}_{(t)}\right) \\
\mathbf{z}^{[2]}_{(t)} &= z^{[2]}\left(\mathbf{W}^{[2]}, \mathbf{x}^{[1]}_{(t)}, \mathbf{b}^{[2]}\right) &&= \mathbf{W}^{[2]}\mathbf{x}^{[1]}_{(t)} + \mathbf{b}^{[2]} \\
\mathbf{x}^{[2]}_{(t)} &= f^{[2]}\left(\mathbf{z}^{[2]}_{(t)}\right) &&= softmax\left(\mathbf{z}^{[2]}_{(t)}\right) \\
e_{(t)} &= E\left(\mathbf{x}^{[2]}_{(t)}, \mathbf{t}_{(t)}\right) &&= -\mathbf{t}_{(t)} \bullet \log \mathbf{x}^{[2]}_{(t)}
\end{aligned}$$

---

b) Write down the model equations for backward propagation.

Solution:

Since we are working with a many to many network each timestep yields an error that will be propagated backwards. For the output layer, the deltas are independent of other timesteps. Applying the chain–rule we get:

$$\begin{aligned}
\delta^{[2]}_{(t)} &= \frac{\partial E}{\partial \mathbf{x}^{[2]}_{(t)}} \frac{\partial \mathbf{x}^{[2]}_{(t)}}{\mathbf{z}^{[2]}_{(t)}} \\
&= \mathbf{x}^{[2]}_{(t)} - \mathbf{t}_{(t)}
\end{aligned}$$

For the last timestep T, the delta for the hidden layer is independent of all other timesteps. Using the chain-rule we get:

$$\begin{aligned}
\delta^{[1]}_{(T)} &= \delta^{[2]}_{(T)} \frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{x}^{[1]}_{(T)}} \frac{\partial \mathbf{x}^{[1]}_{(T)}}{\partial \mathbf{z}^{[1]}_{(T)}} \\
&= \left(\frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{x}^{[1]}_{(T)}} \right)^T \bullet \delta^{[2]}_{(T)} \circ \frac{\partial \mathbf{x}^{[1]}_{(T)}}{\partial \mathbf{z}^{[1]}_{(T)}} \\
&= \left(\mathbf{W}^{[2]} \right)^T \bullet \delta^{[2]}_{(T)} \circ \left(1 - \tanh^2 \mathbf{z}^{[1]}_{(T)} \right)
\end{aligned}$$

For all other timesteps $t \in \{1, \ldots, T-1\}$, the delta for the hidden layer depends on the next timestep's hidden layer. Using the chain-rule we get:

$$\begin{aligned}
\delta^{[1]}_{(t)} &= \left(\delta^{[2]}_{(t)} \frac{\partial \mathbf{z}^{[2]}_{(t)}}{\partial \mathbf{x}^{[1]}_{(t)}} + \delta^{[1]}_{(t+1)} \frac{\partial \mathbf{z}^{[1]}_{(t+1)}}{\partial \mathbf{x}^{[1]}_{(t)}} \right) \frac{\partial \mathbf{x}^{[1]}_{(t)}}{\partial \mathbf{z}^{[1]}_{(t)}} \\
&= \left(\left(\frac{\partial \mathbf{z}^{[2]}_{(t)}}{\partial \mathbf{x}^{[1]}_{(t)}} \right)^T \bullet \delta^{[2]}_{(t)} + \left(\frac{\partial \mathbf{z}^{[1]}_{(t+1)}}{\partial \mathbf{x}^{[1]}_{(t)}} \right)^T \bullet \delta^{[1]}_{(t+1)} \right) \circ \frac{\partial \mathbf{x}^{[1]}_{(t)}}{\partial \mathbf{z}^{[1]}_{(t)}} \\
&= \left(\left(\mathbf{W}^{[2]} \right)^T \bullet \delta^{[2]}_{(t)} + \mathbf{H}^T \bullet \delta^{[1]}_{(t+1)} \right) \circ \left(1 - \tanh^2 \mathbf{z}^{[1]}_{(t)} \right)
\end{aligned}$$

---

c) Perform a stochastic gradient descent update for:

$$\mathbf{x}^{(1)} = \left[\begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} \right]$$

With targets:

$$t^{(1)} = \left[\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right]$$

Initialize all weights and biases to 0.1, using 3 units per hidden layer, initializing the hidden state to all zeros and using $\eta = 1.0$.

Solution:
Initializing the parameters, we have:

$$\mathbf{W}^{[1]} = \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[1]} = \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{H}^{[1]} = \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix}$$

$$\mathbf{W}^{[2]} = \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[2]} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{x}_{(0)}^{[1]} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Now, we can use **a)** to apply forward propagation. Let us start with timestep $t = 1$:

$$\begin{aligned}
\mathbf{z}_{(1)}^{[1]} &= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \\
\mathbf{x}_{(1)}^{[1]} &= \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \\
\mathbf{z}_{(1)}^{[2]} &= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh 0.5 + 0.1 \\ 0.3 \tanh 0.5 + 0.1 \end{pmatrix} \\
\mathbf{x}_{(1)}^{[2]} &= softmax \begin{pmatrix} 0.3 \tanh 0.5 + 0.1 \\ 0.3 \tanh 0.5 + 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}
\end{aligned}$$

Now we can apply the same logic to the second timestep.

$$\begin{aligned}
\mathbf{z}_{(2)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[1]} &= \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{z}_{(2)}^{[2]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \\ 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[2]} &= softmax \begin{pmatrix} 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \\ 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}
\end{aligned}$$

After finishing forward propagation, we can use **b)** to perform backward propagation. For the output layers we have:

$$\begin{aligned}
\delta_{(2)}^{[2]} &= \mathbf{x}_{(2)}^{[2]} - \mathbf{t}_{(2)} &&= \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \\
\delta_{(1)}^{[2]} &= \mathbf{x}_{(1)}^{[2]} - \mathbf{t}_{(1)} &&= \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix}
\end{aligned}$$

For the hidden layer of the last timestep we have:

$$\begin{aligned}
\delta_{(2)}^{[1]} &= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(2)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(2)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1\ 0.1 \\ 0.1\ 0.1 \\ 0.1\ 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}$$

For the hidden layer of the first timestep we have:

$$
\begin{aligned}
\delta_{(1)}^{[1]} &= \left(\left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(1)}^{[2]} + \left(\mathbf{H}^{[1]}\right)^T \bullet \delta_{(2)}^{[1]}\right) \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right) \\
&= \left(\begin{pmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \\ 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\right) \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \left(\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\right) \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

Having the deltas, we can compute the specific gradients for all parameters. Each gradient corresponds to the sum accross all backward paths that lead to that parameter.

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{H}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{H}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(0)}^{[0]}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[1]}\right)^T \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \tanh 0.5 & \tanh 0.5 & \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{W}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{W}^{[1]}}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{W}^{[1]}}\right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[0]}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(2)}^{[0]}\right)^T \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 4 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 8 & 2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{b}^{[1]}} + \delta_{(2)}^{[1]} \circ \frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \mathbf{1} + \delta_{(2)}^{[1]} \circ \mathbf{1} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[2]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[2]} \frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{W}^{[2]}} \\
&= \delta_{(1)}^{[2]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[2]}}{\partial \mathbf{W}^{[2]}} \right)^T + \delta_{(2)}^{[2]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{W}^{[2]}} \right)^T \\
&= \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \right)^T + \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \right)^T \\
&= \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} \bullet \begin{pmatrix} \tanh 0.5 & \tanh 0.5 & \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \bullet \begin{pmatrix} \tanh (1.1 + 0.3 \tanh 0.5) & \tanh (1.1 + 0.3 \tanh \end{pmatrix} \\
&= \begin{pmatrix} -0.5 \tanh 0.5 & -0.5 \tanh 0.5 & -0.5 \tanh 0.5 \\ 0.5 \tanh 0.5 & 0.5 \tanh 0.5 & 0.5 \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.5 \tanh (1.1 + 0.3 \tanh 0.5) & 0.5 \tanh (1.1 + 0.3 \\ -0.5 \tanh (1.1 + 0.3 \tanh 0.5) & -0.5 \tanh (1.1 + 0.3 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 (\tanh (1.1 + 0.3 \tanh 0.5) - \tanh 0.5) & 0.5 (\tanh (1.1 + 0.3 \tanh 0.5) - \tanh 0.5) & 0.5 (\tanh (1.1 + \\ 0.5 (\tanh 0.5 - \tanh (1.1 + 0.3 \tanh 0.5)) & 0.5 (\tanh 0.5 - \tanh (1.1 + 0.3 \tanh 0.5)) & 0.5 (\tanh 0.5 - \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[2]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[2]} \frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\
&= \delta_{(1)}^{[2]} \circ \frac{\partial \mathbf{z}_{(1)}^{[2]}}{\partial \mathbf{b}^{[2]}} + \delta_{(2)}^{[2]} \circ \frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\
&= \delta_{(1)}^{[2]} \circ \mathbf{1} + \delta_{(2)}^{[2]} \circ \mathbf{1} \\
&= \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

Having the gradients, we can perform the stochastic gradient descent updates for all parameters:

$$
\begin{aligned}
\mathbf{H}^{[1]} &= \mathbf{H}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{H}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \\
\mathbf{W}^{[1]} &= \mathbf{W}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix} \\
\mathbf{b}^{[1]} &= \mathbf{b}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[1]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
\mathbf{W}^{[2]} &= \mathbf{W}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[2]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0.5\left(\tanh\left(1.1 + 0.3\tanh 0.5\right) - \tanh 0.5\right) & 0.5\left(\tanh\left(1.1 + 0.3\tanh 0.5\right) - \tanh 0.5 \right. \\ 0.5\left(\tanh 0.5 - \tanh\left(1.1 + 0.3\tanh 0.5\right)\right) & 0.5\left(\tanh 0.5 - \tanh\left(1.1 + 0.3\tanh 0.5\right)\right. \end{pmatrix} \\
&= \begin{pmatrix} 0.1 - 0.5\left(\tanh\left(1.1 + 0.3\tanh 0.5\right) - \tanh 0.5\right) & 0.1 - 0.5\left(\tanh\left(1.1 + 0.3\tanh 0.5\right) - \tanh 0.5\right) & 0.1 - \\ 0.1 - 0.5\left(\tanh 0.5 - \tanh\left(1.1 + 0.3\tanh 0.5\right)\right) & 0.1 - 0.5\left(\tanh 0.5 - \tanh\left(1.1 + 0.3\tanh 0.5\right)\right) & 0.1 - \end{pmatrix} \\
&= \begin{pmatrix} -0.0915 & -0.0915 & -0.0915 \\ 0.2915 & 0.2915 & 0.2915 \end{pmatrix} \\
\mathbf{b}^{[2]} &= \mathbf{b}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[2]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}
\end{aligned}
$$

---

2) Consider a recurrent network with one hidden layer to solve a **many-to-many** task. Take into account that:

- The recurrent connections happen between the hidden layers
- The hidden activation function is the hyperbolic tangent
- The output activation function is the identity
- The error function is the half squared error between output and target

a) Write down the model equations for forward propagation.

Solution:

For all $t \in \{1, \ldots, T\}$:

$$
\begin{aligned}
\mathbf{z}_{(t)}^{[1]} &= z^{[1]}\left(\mathbf{H}^{[1]}, \mathbf{x}_{(t-1)}^{[1]}, \mathbf{W}^{[1]}, \mathbf{x}_{(t)}^{[0]}, \mathbf{b}^{[1]}\right) &&= \mathbf{H}^{[1]}\mathbf{x}_{(t-1)}^{[1]} + \mathbf{W}^{[1]}\mathbf{x}_{(t)}^{[0]} + \mathbf{b}^{[1]} \\
\mathbf{x}_{(t)}^{[1]} &= f^{[1]}\left(\mathbf{z}_{(t)}^{[1]}\right) &&= \tanh\left(\mathbf{z}_{(t)}^{[1]}\right) \\
\mathbf{z}_{(t)}^{[2]} &= z^{[2]}\left(\mathbf{W}^{[2]}, \mathbf{x}_{(t)}^{[1]}, \mathbf{b}^{[2]}\right) &&= \mathbf{W}^{[2]}\mathbf{x}_{(t)}^{[1]} + \mathbf{b}^{[2]} \\
\mathbf{x}_{(t)}^{[2]} &= f^{[2]}\left(\mathbf{z}_{(t)}^{[2]}\right) &&= \mathbf{z}_{(t)}^{[2]} \\
e_{(t)} &= E\left(\mathbf{x}_{(t)}^{[2]}, \mathbf{t}_{(t)}\right) &&= \frac{1}{2}\left\|\mathbf{t}_{(t)} - \mathbf{x}_{(t)}^{[2]}\right\|_2^2
\end{aligned}
$$

---

b) Write down the model equations for backward propagation.

Solution:

Since we are working with a many to many network each timestep yields an error that will be propagated backwards. For the output layer, the deltas are independent of other timesteps. Applying the chain–rule we get:

$$
\begin{aligned}
\delta_{(t)}^{[2]} &= \frac{\partial E}{\partial \mathbf{x}_{(t)}^{[2]}} \frac{\partial \mathbf{x}_{(t)}^{[2]}}{\mathbf{z}_{(t)}^{[2]}} \\
&= \left(\mathbf{x}_{(t)}^{[2]} - \mathbf{t}_{(t)}\right) \circ \mathbf{1} \\
&= \mathbf{x}_{(t)}^{[2]} - \mathbf{t}_{(t)}
\end{aligned}
$$

For the last timestep T, the delta for the hidden layer is independent of all other timesteps. Using the chain-rule we get:

$$
\begin{aligned}
\delta_{(T)}^{[1]} &= \delta_{(T)}^{[2]} \frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{x}_{(T)}^{[1]}} \frac{\partial \mathbf{x}_{(T)}^{[1]}}{\partial \mathbf{z}_{(T)}^{[1]}} \\
&= \left(\frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{x}_{(T)}^{[1]}}\right)^T \bullet \delta_{(T)}^{[2]} \circ \frac{\partial \mathbf{x}_{(T)}^{[1]}}{\partial \mathbf{z}_{(T)}^{[1]}} \\
&= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(T)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(T)}^{[1]}\right)
\end{aligned}
$$

For all other timesteps $t \in \{1, \ldots, T-1\}$, the delta for the hidden layer depends on the next timestep's hidden layer. Using the chain-rule we get:

$$\begin{aligned}
\delta_{(t)}^{[1]} &= \left(\delta_{(t)}^{[2]} \frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{x}_{(t)}^{[1]}} + \delta_{(t+1)}^{[1]} \frac{\partial \mathbf{z}_{(t+1)}^{[1]}}{\partial \mathbf{x}_{(t)}^{[1]}} \right) \frac{\partial \mathbf{x}_{(t)}^{[1]}}{\partial \mathbf{z}_{(t)}^{[1]}} \\
&= \left(\left(\frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{x}_{(t)}^{[1]}} \right)^T \bullet \delta_{(t)}^{[2]} + \left(\frac{\partial \mathbf{z}_{(t+1)}^{[1]}}{\partial \mathbf{x}_{(t)}^{[1]}} \right)^T \bullet \delta_{(t+1)}^{[1]} \right) \circ \frac{\partial \mathbf{x}_{(t)}^{[1]}}{\partial \mathbf{z}_{(t)}^{[1]}} \\
&= \left(\left(\mathbf{W}^{[2]} \right)^T \bullet \delta_{(t)}^{[2]} + \mathbf{H}^T \bullet \delta_{(t+1)}^{[1]} \right) \circ \left(1 - \tanh^2 \mathbf{z}_{(t)}^{[1]} \right)
\end{aligned}$$

---

c) Perform a stochastic gradient descent update for:

$$x^{(1)} = \left[\begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} \right]$$

With targets:

$$t^{(1)} = \left[\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right]$$

Initialize all weights and biases to 0.1, using 3 units per hidden layer, initializing the hidden state to all zeros and using $\eta = 1.0$.

Solution:
Initializing the parameters, we have:

$$\mathbf{W}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[1]} = \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{H}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{W}^{[2]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[2]} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{x}_{(0)}^{[1]} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Now, we can use **a)** to apply forward propagation. Let us start with timestep $t = 1$:

$$\begin{aligned}
\mathbf{z}_{(1)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \\
\mathbf{x}_{(1)}^{[1]} &= \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \\
\mathbf{z}_{(1)}^{[2]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh 0.5 + 0.1 \\ 0.3 \tanh 0.5 + 0.1 \end{pmatrix} \\
\mathbf{x}_{(1)}^{[2]} &= \begin{pmatrix} 0.3 \tanh 0.5 + 0.1 \\ 0.3 \tanh 0.5 + 0.1 \end{pmatrix}
\end{aligned}$$

Now we can apply the same logic to the second timestep.

$$\begin{aligned}
\mathbf{z}_{(2)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[1]} &= \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{z}_{(2)}^{[2]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \\ 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[2]} &= \begin{pmatrix} 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \\ 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \end{pmatrix}
\end{aligned}$$

After finishing forward propagation, we can use **b)** to perform backward propagation. For the output layers we have:

$$\begin{aligned}
\delta_{(2)}^{[2]} = \mathbf{x}_{(2)}^{[2]} - \mathbf{t}_{(2)} &= \begin{pmatrix} 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) + 0.1 \\ 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) + 0.1 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) + 0.1 \\ 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.9 \end{pmatrix} \\
&= \begin{pmatrix} o_{21} \\ o_{22} \end{pmatrix} \\
\delta_{(1)}^{[2]} = \mathbf{x}_{(1)}^{[2]} - \mathbf{t}_{(1)} &= \begin{pmatrix} 0.3\tanh 0.5 + 0.1 \\ 0.3\tanh 0.5 + 0.1 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.3\tanh 0.5 - 0.9 \\ 0.3\tanh 0.5 + 0.1 \end{pmatrix} \\
&= \begin{pmatrix} o_{11} \\ o_{12} \end{pmatrix}
\end{aligned}$$

For the hidden layer of the last timestep we have:

$$\begin{aligned}
\delta_{(2)}^{[1]} &= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(2)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(2)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \\ 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) + 0.1 \\ 0.3\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.9 \end{pmatrix} \circ \\
&\circ \begin{pmatrix} 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \\ 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \\ 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} 0.06\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.08 \\ 0.06\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.08 \\ 0.06\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.08 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \\ 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \\ 1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix}
\end{aligned}$$

Where:

$$h_2 = \left(0.06\tanh\left(1.1 + 0.3\tanh 0.5\right) - 0.08\right)\left(1 - \tanh^2\left(1.1 + 0.3\tanh 0.5\right)\right)$$

For the hidden layer of the first timestep we have:

$$\delta_{(1)}^{[1]} = \left(\left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(1)}^{[2]} + \left(\mathbf{H}^{[1]}\right)^T \bullet \delta_{(2)}^{[1]}\right) \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right)$$

$$\left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(1)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right) + \left(\mathbf{H}^{[1]}\right)^T \bullet \delta_{(2)}^{[1]} \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right)$$

$$= \begin{pmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \\ 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0.3\tanh 0.5 - 0.9 \\ 0.3\tanh 0.5 + 0.1 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} +$$

$$+ \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \bullet$$

$$\bullet \begin{pmatrix} (0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \\ (0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \\ (0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \end{pmatrix} \circ$$

$$\circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} 0.06\tanh 0.5 - 0.08 \\ 0.06\tanh 0.5 - 0.08 \\ 0.06\tanh 0.5 - 0.08 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} +$$

$$+ \begin{pmatrix} 0.3\,(0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \\ 0.3\,(0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \\ 0.3\,(0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right) \end{pmatrix} \circ$$

$$\circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix}$$

$$= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix}$$

Where:

$$h_1 = (0.06\tanh 0.5 - 0.08)\left(1 - \tanh^2 0.5\right) +$$

$$+ \left(0.3\,(0.06\tanh(1.1 + 0.3\tanh 0.5) - 0.08)\left(1 - \tanh^2(1.1 + 0.3\tanh 0.5)\right)\right)\left(1 - \tanh^2 0.5\right)$$

Having the deltas, we can compute the specific gradients for all parameters. Each gradient corresponds to the sum accross all backward paths that lead to that parameter.

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{H}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{H}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{H}^{[1]}} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{H}^{[1]}} \right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(0)}^{[0]} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[1]} \right)^T \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \begin{pmatrix} \tanh 0.5 & \tanh 0.5 & \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{W}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[0]} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(2)}^{[0]} \right)^T \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} \begin{pmatrix} 4 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \begin{pmatrix} 0 & 8 & 2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 4h_1 & 0 & 0 & 0 \\ 4h_1 & 0 & 0 & 0 \\ 4h_1 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 8h_2 & 2h_2 & 0 \\ 0 & 8h_2 & 2h_2 & 0 \\ 0 & 8h_2 & 2h_2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \end{pmatrix}
\end{aligned}
$$

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{b}^{[1]}} + \delta_{(2)}^{[1]} \circ \frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \mathbf{1} + \delta_{(2)}^{[1]} \circ \mathbf{1} \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \\
&= \begin{pmatrix} h_1 + h_2 \\ h_1 + h_2 \\ h_1 + h_2 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[2]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[2]} \frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{W}^{[2]}} \\
&= \delta_{(1)}^{[2]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[2]}}{\partial \mathbf{W}^{[2]}} \right)^T + \delta_{(2)}^{[2]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{W}^{[2]}} \right)^T \\
&= \begin{pmatrix} o_{11} \\ o_{12} \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \right)^T + \begin{pmatrix} o_{21} \\ o_{22} \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \right)^T \\
&= \begin{pmatrix} o_{11} \\ o_{12} \end{pmatrix} \bullet \begin{pmatrix} \tanh 0.5 & \tanh 0.5 & \tanh 0.5 \end{pmatrix} + \begin{pmatrix} o_{21} \\ o_{22} \end{pmatrix} \bullet \begin{pmatrix} \tanh (1.1 + 0.3 \tanh 0.5) & \tanh (1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} o_{11} \tanh 0.5 & o_{11} \tanh 0.5 & o_{11} \tanh 0.5 \\ o_{12} \tanh 0.5 & o_{12} \tanh 0.5 & o_{12} \tanh 0.5 \end{pmatrix} + \begin{pmatrix} o_{21} \tanh (1.1 + 0.3 \tanh 0.5) & o_{21} \tanh (1.1 + 0.3 \tanh 0. \\ o_{22} \tanh (1.1 + 0.3 \tanh 0.5) & o_{22} \tanh (1.1 + 0.3 \tanh 0. \end{pmatrix} \\
&= \begin{pmatrix} w_{21} & w_{21} & w_{21} \\ w_{22} & w_{22} & w_{22} \end{pmatrix}
\end{aligned}$$

Where:

$w_{21} = o_{11} \tanh 0.5 + o_{21} \tanh (1.1 + 0.3 \tanh 0.5)$

$w_{22} = o_{12} \tanh 0.5 + o_{22} \tanh (1.1 + 0.3 \tanh 0.5)$

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[2]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[2]} \frac{\partial \mathbf{z}_{(t)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\
&= \delta_{(1)}^{[2]} \circ \frac{\partial \mathbf{z}_{(1)}^{[2]}}{\partial \mathbf{b}^{[2]}} + \delta_{(2)}^{[1]} \circ \frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\
&= \delta_{(1)}^{[2]} \circ \mathbf{1} + \delta_{(2)}^{[2]} \circ \mathbf{1} \\
&= \begin{pmatrix} o_{11} \\ o_{12} \end{pmatrix} + \begin{pmatrix} o_{21} \\ o_{22} \end{pmatrix} \\
&= \begin{pmatrix} o_{11} + o_{21} \\ o_{12} + o_{22} \end{pmatrix}
\end{aligned}$$

Having the gradients, we can perform the stochastic gradient descent updates for all parameters:

$$\begin{aligned}
\mathbf{H}^{[1]} &= \mathbf{H}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{H}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0.1039 & 0.1039 & 0.1039 \\ 0.1039 & 0.1039 & 0.1039 \\ 0.1039 & 0.1039 & 0.1039 \end{pmatrix} \\
\mathbf{W}^{[1]} &= \mathbf{W}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.2723 & 0.1669 & 0.1167 & 0.1 \\ 0.2723 & 0.1669 & 0.1167 & 0.1 \\ 0.2723 & 0.1669 & 0.1167 & 0.1 \end{pmatrix} \\
\mathbf{b}^{[1]} &= \mathbf{b}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[1]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} h_1 + h_2 \\ h_1 + h_2 \\ h_1 + h_2 \end{pmatrix} \\
&= \begin{pmatrix} 0.1515 \\ 0.1515 \\ 0.1515 \end{pmatrix} \\
\mathbf{W}^{[2]} &= \mathbf{W}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[2]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} w_{21} & w_{21} & w_{21} \\ w_{22} & w_{22} & w_{22} \end{pmatrix} \\
&= \begin{pmatrix} 0.1531 & 0.1531 & 0.1531 \\ 0.5360 & 0.5360 & 0.5360 \end{pmatrix} \\
\mathbf{b}^{[2]} &= \mathbf{b}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[2]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} o_{11} + o_{21} \\ o_{12} + o_{22} \end{pmatrix} \\
&= \begin{pmatrix} 0.5078 \\ 0.5078 \end{pmatrix}
\end{aligned}$$

---

3) Consider a recurrent network with one hidden layer to solve a **many-to-one** task. Take into account that:

- The recurrent connections happen between the hidden layers
- The hidden activation function is the hyperbolic tangent
- The output activation function is the softmax
- The error function is the cross-entropy between output and target

a) Write down the model equations for forward propagation.

Solution:

Timesteps before the last $t \in \{1, \ldots, T-1\}$ are only processed until the hidden layer:

$$\mathbf{z}_{(t)}^{[1]} = \mathbf{H}^{[1]}\mathbf{x}_{(t-1)}^{[1]} + \mathbf{W}^{[1]}\mathbf{x}_{(t)}^{[0]} + \mathbf{b}^{[1]}$$

$$\mathbf{x}_{(t)}^{[1]} = \tanh\left(\mathbf{z}_{(t)}^{[1]}\right)$$

The last timestep receives the hidden state from the previous timestep and performs the full network operation until the output:

$$\mathbf{z}_{(T)}^{[1]} = \mathbf{H}^{[1]}\mathbf{x}_{(T-1)}^{[1]} + \mathbf{W}^{[1]}\mathbf{x}_{(T)}^{[0]} + \mathbf{b}^{[1]}$$

$$\mathbf{x}_{(T)}^{[1]} = \tanh\left(\mathbf{z}_{(T)}^{[1]}\right)$$

$$\mathbf{z}_{(T)}^{[2]} = \mathbf{W}^{[2]}\mathbf{x}_{(T)}^{[1]} + \mathbf{b}^{[2]}$$

$$\mathbf{x}_{(T)}^{[2]} = softmax\left(\mathbf{z}_{(T)}^{[2]}\right)$$

$$e = -\mathbf{t} \bullet \log \mathbf{x}_{(T)}^{[2]}$$

---

b) Write down the model equations for backward propagation.

Solution:

Since we are working with a many to one network only the last timestep will have a delta for the output layer. So, applying the chain–rule we get:

$$\begin{aligned}\delta_{(T)}^{[2]} &= \frac{\partial E}{\partial \mathbf{x}_{(T)}^{[2]}} \frac{\partial \mathbf{x}_{(T)}^{[2]}}{\mathbf{z}_{(T)}^{[2]}} \\ &= \mathbf{x}_{(T)}^{[2]} - \mathbf{t}_{(T)}\end{aligned}$$

For the last timestep T, the delta for the hidden layer is independent of all other timesteps. Using the chain-rule we get:

$$\begin{aligned}\delta_{(T)}^{[1]} &= \delta_{(T)}^{[2]} \frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{x}_{(T)}^{[1]}} \frac{\partial \mathbf{x}_{(T)}^{[1]}}{\partial \mathbf{z}_{(T)}^{[1]}} \\ &= \left(\frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{x}_{(T)}^{[1]}}\right)^{T} \bullet \delta_{(T)}^{[2]} \circ \frac{\partial \mathbf{x}_{(T)}^{[1]}}{\partial \mathbf{z}_{(T)}^{[1]}} \\ &= \left(\mathbf{W}^{[2]}\right)^{T} \bullet \delta_{(T)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(T)}^{[1]}\right)\end{aligned}$$

For all other timesteps $t \in \{1, \ldots, T-1\}$, the delta for the hidden layer depends only on the next timestep's hidden layer. Using the chain-rule we get:

$$\begin{aligned}
\delta_{(t)}^{[1]} &= \delta_{(t+1)}^{[1]} \frac{\partial \mathbf{z}_{(t+1)}^{[1]}}{\partial \mathbf{x}_{(t)}^{[1]}} \frac{\partial \mathbf{x}_{(t)}^{[1]}}{\partial \mathbf{z}_{(t)}^{[1]}} \\
&= \left(\frac{\partial \mathbf{z}_{(t+1)}^{[1]}}{\partial \mathbf{x}_{(t)}^{[1]}} \right)^T \bullet \delta_{(t+1)}^{[1]} \circ \frac{\partial \mathbf{x}_{(t)}^{[1]}}{\partial \mathbf{z}_{(t)}^{[1]}} \\
&= \left(\mathbf{H}^{[1]} \right)^T \bullet \delta_{(t+1)}^{[1]} \circ \left(1 - \tanh^2 \mathbf{z}_{(t)}^{[1]} \right)
\end{aligned}$$

---

c) Perform a stochastic gradient descent update for:

$$x^{(1)} = \left[\begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} \right]$$

With targets:

$$\mathbf{t} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Initialize all weights and biases to 0.1, using 3 units per hidden layer, initializing the hidden state to all zeros and using $\eta = 1.0$.

Solution:
Initializing the parameters, we have:

$$\mathbf{W}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[1]} = \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{H}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{W}^{[2]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[2]} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{x}_{(0)}^{[1]} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Now, we can use **a)** to apply forward propagation. Let us start with timestep $t = 1$:

$$\begin{aligned}
\mathbf{z}_{(1)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} \\
\mathbf{x}_{(1)}^{[1]} &= \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}
\end{aligned}$$

Now we can apply the same logic to the second timestep.

$$\begin{aligned}
\mathbf{z}_{(2)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[1]} &= \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{z}_{(2)}^{[2]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \\ 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[2]} &= softmax \begin{pmatrix} 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \\ 0.3 \tanh (1.1 + 0.3 \tanh 0.5) + 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}
\end{aligned}$$

After finishing forward propagation, we can use **b)** to perform backward propagation. For the output layer in the last timestep we have:

$$\delta_{(2)}^{[2]} = \mathbf{x}_{(2)}^{[2]} - \mathbf{t}_{(2)} \quad = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix}$$

For the hidden layer of the last timestep we have:

$$
\begin{aligned}
\delta_{(2)}^{[1]} &= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(2)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(2)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \\ 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \\ 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \\ 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \\ 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \\ 1 - \tanh^2 (1.1 + 0.3 \tanh 0.5) \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

For the hidden layer of the first timestep we have:

$$
\begin{aligned}
\delta_{(1)}^{[1]} &= \left(\mathbf{H}^{[1]}\right)^T \bullet \delta_{(2)}^{[1]} \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} \bullet \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

Having the deltas, we can compute the specific gradients for all parameters. Each gradient corresponds to the sum accross all backward paths that lead to that parameter.

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{H}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{H}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(0)}^{[0]}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[1]}\right)^T \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} \tanh 0.5 & \tanh 0.5 & \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{W}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[0]} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(2)}^{[0]} \right)^T \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 4 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 8 & 2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{b}^{[1]}} + \delta_{(2)}^{[1]} \circ \frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \mathbf{1} + \delta_{(2)}^{[1]} \circ \mathbf{1} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}
\end{aligned}
$$

For the output layer parameters there is no sum because, unlike the many-to-many network, this network only has an output layer for the last timestep:

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[2]}} &= \delta_{(T)}^{[2]} \frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{W}^{[2]}} \\
&= \delta_{(2)}^{[2]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{W}^{[2]}} \right)^T \\
&= \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \right)^T \\
&= \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \bullet \begin{pmatrix} d & d & d \end{pmatrix} \\
&= \begin{pmatrix} 0.5d & 0.5d & 0.5d \\ -0.5d & -0.5d & -0.5d \end{pmatrix}
\end{aligned}
$$

Where $d = \tanh(1.1 + 0.3 \tanh 0.5)$.

$$\begin{aligned}\frac{\partial E}{\partial \mathbf{b}^{[2]}} &= \delta_{(T)}^{[2]} \frac{\partial \mathbf{z}_{(T)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\ &= \delta_{(2)}^{[2]} \circ \frac{\partial \mathbf{z}_{(2)}^{[2]}}{\partial \mathbf{b}^{[2]}} \\ &= \delta_{(2)}^{[2]} \circ \mathbf{1} \\ &= \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix}\end{aligned}$$

Having the gradients, we can perform the stochastic gradient descent updates for all parameters:

$$\begin{aligned}\mathbf{H}^{[1]} &= \mathbf{H}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{H}^{[1]}} \\ &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0\ 0\ 0 \\ 0\ 0\ 0 \\ 0\ 0\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\mathbf{W}^{[1]} &= \mathbf{W}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[1]}} \\ &= \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\mathbf{b}^{[1]} &= \mathbf{b}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[1]}} \\ &= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\mathbf{W}^{[2]} &= \mathbf{W}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[2]}} \\ &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0.5d & 0.5d & 0.5d \\ -0.5d & -0.5d & -0.5d \end{pmatrix} \\ &= \begin{pmatrix} 0.1 - 0.5d\ 0.1 - 0.5d\ 0.1 - 0.5d \\ 0.1 + 0.5d\ 0.1 + 0.5d\ 0.1 + 0.5d \end{pmatrix} \\ &= \begin{pmatrix} -0.3225 & -0.3225 & -0.3225 \\ 0.5225 & 0.5225 & 0.5225 \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\mathbf{b}^{[2]} &= \mathbf{b}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[2]}} \\ &= \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} 0.5 \\ -0.5 \end{pmatrix} \\ &= \begin{pmatrix} -0.4 \\ 0.6 \end{pmatrix}\end{aligned}$$

---

4) Consider a recurrent network with one hidden layer to solve a **many-to-one** task. Take into account that:

- The recurrent connections happen between the hidden layers
- The hidden activation function is the hyperbolic tangent
- The output activation function is the identity
- The error function is the half squared error between output and target

a) Write down the model equations for forward propagation.

Solution:

Timesteps before the last $t \in \{1, \ldots, T-1\}$ are only processed until the hidden layer:

$$\mathbf{z}^{[1]}_{(t)} = \mathbf{H}^{[1]}\mathbf{x}^{[1]}_{(t-1)} + \mathbf{W}^{[1]}\mathbf{x}^{[0]}_{(t)} + \mathbf{b}^{[1]}$$

$$\mathbf{x}^{[1]}_{(t)} = \tanh\left(\mathbf{z}^{[1]}_{(t)}\right)$$

The last timestep receives the hidden state from the previous timestep and performs the full network operation until the output:

$$\mathbf{z}^{[1]}_{(T)} = \mathbf{H}^{[1]}\mathbf{x}^{[1]}_{(T-1)} + \mathbf{W}^{[1]}\mathbf{x}^{[0]}_{(T)} + \mathbf{b}^{[1]}$$

$$\mathbf{x}^{[1]}_{(T)} = \tanh\left(\mathbf{z}^{[1]}_{(T)}\right)$$

$$\mathbf{z}^{[2]}_{(T)} = \mathbf{W}^{[2]}\mathbf{x}^{[1]}_{(T)} + \mathbf{b}^{[2]}$$

$$\mathbf{x}^{[2]}_{(T)} = \mathbf{z}^{[2]}_{(T)}$$

$$e = -\mathbf{t} \bullet \log \mathbf{x}^{[2]}_{(T)}$$

---

b) Write down the model equations for backward propagation.

Solution:

Since we are working with a many to one network only the last timestep will have a delta for the output layer. So, applying the chain–rule we get:

$$\begin{aligned}
\delta^{[2]}_{(T)} &= \frac{\partial E}{\partial \mathbf{x}^{[2]}_{(T)}} \frac{\partial \mathbf{x}^{[2]}_{(T)}}{\mathbf{z}^{[2]}_{(T)}} \\
&= \left(\mathbf{x}^{[2]}_{(T)} - \mathbf{t}_{(T)}\right) \circ \mathbf{1} \\
&= \mathbf{x}^{[2]}_{(T)} - \mathbf{t}_{(T)}
\end{aligned}$$

For the last timestep T, the delta for the hidden layer is independent of all other timesteps. Using the chain-rule we get:

$$\begin{aligned}
\delta^{[1]}_{(T)} &= \delta^{[2]}_{(T)} \frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{x}^{[1]}_{(T)}} \frac{\partial \mathbf{x}^{[1]}_{(T)}}{\partial \mathbf{z}^{[1]}_{(T)}} \\
&= \left(\frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{x}^{[1]}_{(T)}}\right)^T \bullet \delta^{[2]}_{(T)} \circ \frac{\partial \mathbf{x}^{[1]}_{(T)}}{\partial \mathbf{z}^{[1]}_{(T)}} \\
&= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta^{[2]}_{(T)} \circ \left(1 - \tanh^2 \mathbf{z}^{[1]}_{(T)}\right)
\end{aligned}$$

For all other timesteps $t \in \{1, \ldots, T-1\}$, the delta for the hidden layer depends only on the next timestep's hidden layer. Using the chain-rule we get:

$$\begin{aligned}
\delta^{[1]}_{(t)} &= \delta^{[1]}_{(t+1)} \frac{\partial \mathbf{z}^{[1]}_{(t+1)}}{\partial \mathbf{x}^{[1]}_{(t)}} \frac{\partial \mathbf{x}^{[1]}_{(t)}}{\partial \mathbf{z}^{[1]}_{(t)}} \\
&= \left(\frac{\partial \mathbf{z}^{[1]}_{(t+1)}}{\partial \mathbf{x}^{[1]}_{(t)}}\right)^T \bullet \delta^{[1]}_{(t+1)} \circ \frac{\partial \mathbf{x}^{[1]}_{(t)}}{\partial \mathbf{z}^{[1]}_{(t)}} \\
&= \left(\mathbf{H}^{[1]}\right)^T \bullet \delta^{[1]}_{(t+1)} \circ \left(1 - \tanh^2 \mathbf{z}^{[1]}_{(t)}\right)
\end{aligned}$$

---

c) Perform a stochastic gradient descent update for:

$$x^{(1)} = \left[\begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix}\right]$$

With targets:

$$\mathbf{t} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Initialize all weights and biases to 0.1, using 3 units per hidden layer, initializing the hidden state to all zeros and using $\eta = 1.0$.

Solution:
Initializing the parameters, we have:

$$\mathbf{W}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[1]} = \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{H}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{W}^{[2]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix}$$

$$\mathbf{b}^{[2]} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}$$

$$\mathbf{x}_{(0)}^{[1]} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Now, we can use **a)** to apply forward propagation. Let us start with timestep $t = 1$:

$$\mathbf{z}_{(1)}^{[1]} = \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix}$$

$$= \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

$$\mathbf{x}_{(1)}^{[1]} = \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix}$$

Now we can apply the same logic to the second timestep.

$$\begin{aligned}
\mathbf{z}_{(2)}^{[1]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 0.5 \\ 0.5 \\ 0.5 \end{pmatrix} + \begin{pmatrix} 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1\ 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ 8 \\ 2 \\ 0 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[1]} &= \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \\
\mathbf{z}_{(2)}^{[2]} &= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} + \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} \\
&= \begin{pmatrix} 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \\ 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \end{pmatrix} \\
\mathbf{x}_{(2)}^{[2]} &= \begin{pmatrix} 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \\ 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \end{pmatrix}
\end{aligned}$$

After finishing forward propagation, we can use **b)** to perform backward propagation. For the output layer in the last timestep we have:

$$\begin{aligned}
\delta_{(2)}^{[2]} &= \mathbf{x}_{(2)}^{[2]} - \mathbf{t}_{(2)} \\
&= \begin{pmatrix} 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \\ 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} o \\ o - 1 \end{pmatrix}
\end{aligned}$$

Where $o = 0.3 \tanh \left(1.1 + 0.3 \tanh 0.5\right) + 0.1$.

For the hidden layer of the last timestep we have:

$$\begin{aligned}
\delta_{(2)}^{[1]} &= \left(\mathbf{W}^{[2]}\right)^T \bullet \delta_{(2)}^{[2]} \circ \left(1 - \tanh^2 \mathbf{z}_{(2)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1\ 0.1 \\ 0.1\ 0.1 \\ 0.1\ 0.1 \end{pmatrix} \bullet \begin{pmatrix} o \\ o - 1 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} 0.2o - 0.1 \\ 0.2o - 0.1 \\ 0.2o - 0.1 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \\ 1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right) \end{pmatrix} \\
&= \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix}
\end{aligned}$$

Where $h_2 = (0.2o - 0.1)\left(1 - \tanh^2 \left(1.1 + 0.3 \tanh 0.5\right)\right)$.

For the hidden layer of the first timestep we have:

$$
\begin{aligned}
\delta_{(1)}^{[1]} &= \left(\mathbf{H}^{[1]}\right)^T \bullet \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \circ \left(1 - \tanh^2 \mathbf{z}_{(1)}^{[1]}\right) \\
&= \begin{pmatrix} 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.1 \end{pmatrix} \bullet \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0.3h_2 \\ 0.3h_2 \\ 0.3h_2 \end{pmatrix} \circ \begin{pmatrix} 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \\ 1 - \tanh^2 0.5 \end{pmatrix} \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix}
\end{aligned}
$$

Where $h_1 = 0.3h_2 \left(1 - \tanh^2 0.5\right)$.

Having the deltas, we can compute the specific gradients for all parameters. Each gradient corresponds to the sum accross all backward paths that lead to that parameter.

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{H}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{H}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{H}^{[1]}}\right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(0)}^{[0]}\right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[1]}\right)^T \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} \begin{pmatrix} 0\ 0\ 0 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \begin{pmatrix} \tanh 0.5\ \tanh 0.5\ \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} h_2 \tanh 0.5\ h_2 \tanh 0.5\ h_2 \tanh 0.5 \\ h_2 \tanh 0.5\ h_2 \tanh 0.5\ h_2 \tanh 0.5 \\ h_2 \tanh 0.5\ h_2 \tanh 0.5\ h_2 \tanh 0.5 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{W}^{[1]}} \\
&= \delta_{(1)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{W}^{[1]}} \right)^T \\
&= \delta_{(1)}^{[1]} \bullet \left(\mathbf{x}_{(1)}^{[0]} \right)^T + \delta_{(2)}^{[1]} \bullet \left(\mathbf{x}_{(2)}^{[0]} \right)^T \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} \begin{pmatrix} 4 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \begin{pmatrix} 0 & 8 & 2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 4h_1 & 0 & 0 & 0 \\ 4h_1 & 0 & 0 & 0 \\ 4h_1 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 8h_2 & 2h_2 & 0 \\ 0 & 8h_2 & 2h_2 & 0 \\ 0 & 8h_2 & 2h_2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[1]}} &= \sum_{t=1}^{T} \delta_{(t)}^{[1]} \frac{\partial \mathbf{z}_{(t)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \frac{\partial \mathbf{z}_{(1)}^{[1]}}{\partial \mathbf{b}^{[1]}} + \delta_{(2)}^{[1]} \circ \frac{\partial \mathbf{z}_{(2)}^{[1]}}{\partial \mathbf{b}^{[1]}} \\
&= \delta_{(1)}^{[1]} \circ \mathbf{1} + \delta_{(2)}^{[1]} \circ \mathbf{1} \\
&= \begin{pmatrix} h_1 \\ h_1 \\ h_1 \end{pmatrix} + \begin{pmatrix} h_2 \\ h_2 \\ h_2 \end{pmatrix} \\
&= \begin{pmatrix} h_1 + h_2 \\ h_1 + h_2 \\ h_1 + h_2 \end{pmatrix}
\end{aligned}
$$

For the output layer parameters there is no sum because, unlike the many-to-many network, this network only has an output layer for the last timestep:

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{W}^{[2]}} &= \delta^{[2]}_{(T)} \frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{W}^{[2]}} \\
&= \delta^{[2]}_{(2)} \bullet \left(\frac{\partial \mathbf{z}^{[2]}_{(2)}}{\partial \mathbf{W}^{[2]}} \right)^T \\
&= \begin{pmatrix} o \\ o-1 \end{pmatrix} \bullet \left(\tanh \begin{pmatrix} 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \\ 1.1 + 0.3 \tanh 0.5 \end{pmatrix} \right)^T \\
&= \begin{pmatrix} o \\ o-1 \end{pmatrix} \bullet \begin{pmatrix} w & w & w \end{pmatrix} \\
&= \begin{pmatrix} ow & ow & ow \\ (o-1)\, w & (o-1)\, w & (o-1)\, w \end{pmatrix}
\end{aligned}$$

Where $w = \tanh(1.1 + 0.3 \tanh 0.5)$.

$$\begin{aligned}
\frac{\partial E}{\partial \mathbf{b}^{[2]}} &= \delta^{[2]}_{(T)} \frac{\partial \mathbf{z}^{[2]}_{(T)}}{\partial \mathbf{b}^{[2]}} \\
&= \delta^{[2]}_{(2)} \circ \frac{\partial \mathbf{z}^{[2]}_{(2)}}{\partial \mathbf{b}^{[2]}} \\
&= \delta^{[2]}_{(2)} \circ \mathbf{1} \\
&= \begin{pmatrix} o \\ o-1 \end{pmatrix}
\end{aligned}$$

Having the gradients, we can perform the stochastic gradient descent updates for all parameters:

$$\begin{aligned}
\mathbf{H}^{[1]} &= \mathbf{H}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{H}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \\ h_2 \tanh 0.5 & h_2 \tanh 0.5 & h_2 \tanh 0.5 \end{pmatrix} \\
&= \begin{pmatrix} 0.1039 & 0.1039 & 0.1039 \\ 0.1039 & 0.1039 & 0.1039 \\ 0.1039 & 0.1039 & 0.1039 \end{pmatrix}
\end{aligned}$$

$$\begin{aligned}
\mathbf{W}^{[1]} &= \mathbf{W}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[1]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \\ 4h_1 & 8h_2 & 2h_2 & 0 \end{pmatrix} \\
&= \begin{pmatrix} 0.1079 & 0.1669 & 0.1167 & 0.1 \\ 0.1079 & 0.1669 & 0.1167 & 0.1 \\ 0.1079 & 0.1669 & 0.1167 & 0.1 \end{pmatrix}
\end{aligned}$$

$$
\begin{aligned}
\mathbf{b}^{[1]} &= \mathbf{b}^{[1]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[1]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} h_1 + h_2 \\ h_1 + h_2 \\ h_1 + h_2 \end{pmatrix} \\
&= \begin{pmatrix} 0.1104 \\ 0.1104 \\ 0.1104 \end{pmatrix} \\
\mathbf{W}^{[2]} &= \mathbf{W}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{W}^{[2]}} \\
&= \begin{pmatrix} 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 \end{pmatrix} - 1 \begin{pmatrix} ow & ow & ow \\ (o-1)\,w & (o-1)\,w & (o-1)\,w \end{pmatrix} \\
&= \begin{pmatrix} -0.1987 & -0.1987 & -0.1987 \\ 0.6463 & 0.6463 & 0.6463 \end{pmatrix} \\
\mathbf{b}^{[2]} &= \mathbf{b}^{[2]} - \eta \frac{\partial E}{\partial \mathbf{b}^{[2]}} \\
&= \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix} - 1 \begin{pmatrix} o \\ o-1 \end{pmatrix} \\
&= \begin{pmatrix} -0.2535 \\ 0.7465 \end{pmatrix}
\end{aligned}
$$

---

3 Thinking Questions

a) Why is it said that backpropagation through time is very hard to implement in parallel?

b) What factors do you think play a role when choosing a CNNs hyperparameters?

c) Can you see why is it said that a CNN is a smart way to regularize an MLP through a different architecture? Think about some key factors like free parameters, bias and generalization.