Residual connections: a gradient highway

[Deep Residual Learning (https://arxiv.org/abs/1512.03385)](https://arxiv.org/abs/1512.03385)

TL;DR

- We have encountered the Vanishing Gradient problem several times.
- This is a major impediment to training deep (many layers) networks.
- The solution is to give the gradient a path to flow backward undiminished.
- This simple solution is called a Skip or Residual Connection

Consider two layers of a NN

$$\begin{aligned} \mathbf{y}_{(l-1)} &= a_{(l-1)} \left(f_{(l-1)}(\mathbf{y}_{(l-2)}) \right) \\ \mathbf{y}_{(l)} &= a_{(l)} \left(f_{(l)}(\mathbf{y}_{(l-1)}) \right) \end{aligned}$$

Suppose we modify layer $l+1$ by adding $\mathbf{y}_{(l-1)}$ to the output

$$\begin{aligned}
\mathbf{y}_{(l-1)} &= a_{(l-1)} && \left(f_{(l-1)}(\mathbf{y}_{(l-2)})\right) \\
\mathbf{y}_{(l)} &= a_{(l)} && \left(f'_{(l)}(\mathbf{y}_{(l-1)})\right) + \mathbf{y}_{(l-1)}
\end{aligned}$$

If the original and modified 2 layer mini-networks compute the same function from $\mathbf{y}_{(l-1)}$ to $\mathbf{y}_{(l+1)}$

$$
\begin{aligned}
a_{(l)} \left(f_{(l)}(\mathbf{y}_{(l-1)})\right) & = & a_{(l)} \left(f'_{(l)}(\mathbf{y}_{(l-1)})\right) + \mathbf{y}_{(l-1)} \\
a_{(l)} \left(f'_{(l)}(\mathbf{y}_{(l-1)})\right) & = & a_{(l)} \left(f_{(l)}(\mathbf{y}_{(l-1)})\right) - \mathbf{y}_{(l-1)}
\end{aligned}
$$

In other words:

- we have forced the modified second layer to learn the "residual" of the unmodified layer with respect to $\mathbf{y}_{(l-1)}$.

This seems strange (and pointless) until you consider the Back Propagation process.

Recall how the loss gradient

$$\mathcal{L}'_{(l)} = \frac{\partial \mathcal{L}}{\partial y_{(l)}}$$

propagates backwards

$$\mathcal{L}'_{(l-1)} \quad = \quad \mathcal{L}'_{(l)} \frac{\partial \mathbf{y}_{(l)}}{\partial \mathbf{y}_{(l-1)}}$$

In the unmodified mini-NN the local derivative

$$\frac{\partial \mathbf{y}_{(l)}}{\partial \mathbf{y}_{(l-1)}} = \frac{a_{(l)}\left(f_{(l)}(\ldots)\right)}{\partial \mathbf{y}_{(l-1)}}$$

whereas, in the modified mini-NN the local derivative becomes

$$\frac{\partial \mathbf{y}_{(l)}}{\partial \mathbf{y}_{(l-1)}} = \frac{a_{(l)}\left(f'_{(l)}(\ldots)\right)}{\partial \mathbf{y}_{(l-1)}} + 1$$

When $\frac{\partial \mathbf{y}_{(l)}}{\partial \mathbf{y}_{(l-1)}}$ is multiplied by $\mathcal{L}'_{(l)}$ to obtain $\mathcal{L}'_{(l-1)}$

- the "upstream" loss gradient $\mathcal{L}'_{(l-1)}$
- flows backwards to layer $l-1$ unmodulate *because of the* $+1$ term in the modified local derivative.

This simple trick vanquishes the vanishing gradient !

It is one of the major reasons that we are able to train extremely deep NN's.

There is another important implication:

- adding an additional layer cannot result in increased loss

This is because there exists a set of weights $\mathbf{W}_{(l)}$ for which

$$a_{(l)}\left(f'_{(l)}(\mathbf{y}_{(l-1)})\right) = 0$$

This means the modified second layer computes the identity functon

$$\mathbf{y}_{(l)} = \mathbf{y}_{(l-1)}$$

So if adding the second layer had the potential of increasing loss relative to a one layer network

- the optimization would learn the identity function instead, an no increase in loss rsults-

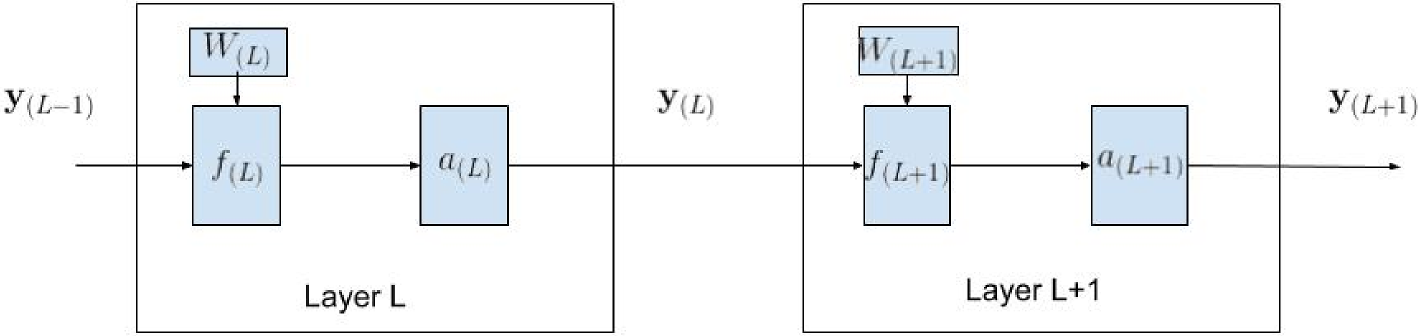
Without the skip connection, it is empircally difficult for a NN to learn an identity function as a layer.

There is an unresolved debate where to place the "head" of the skip connection

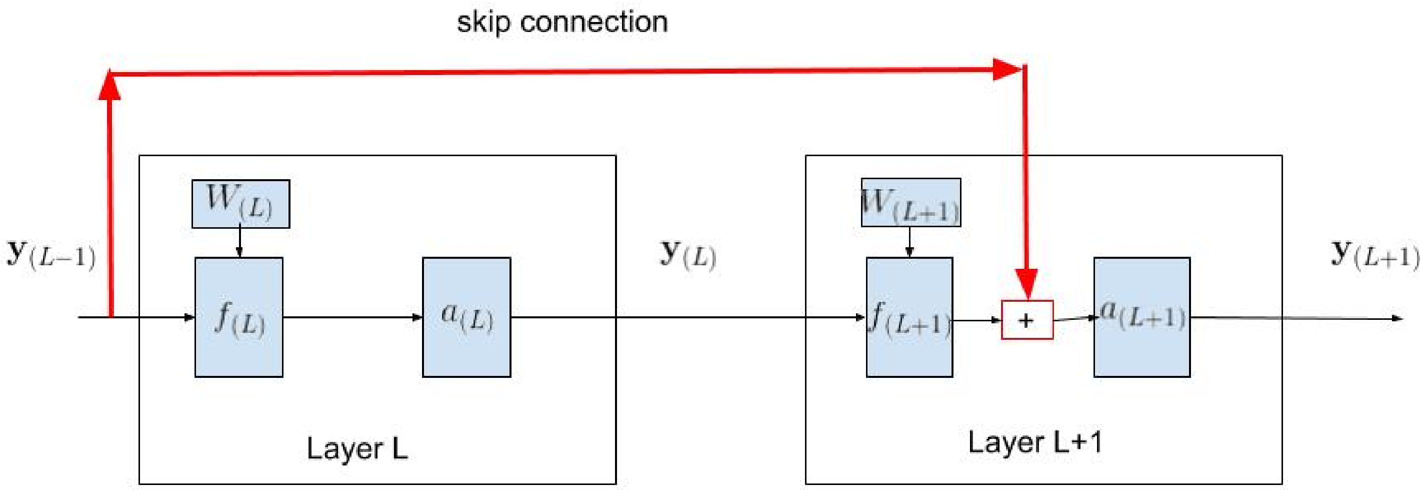
- insider the activation function
- outside the activation function

We choose the latter to simplify the derivative expression for the loss gradient.

"Plain" Neural Network



Neural Network with skip connection



Preview: skip connections in LSTM’s, GRU’s

The gradient highway also turns out to be useful in RNN's.

There are more powerful variants of the RNN called LSTM and GRU which avoid vanishing gradients, partially through the use of skip connections.

These variants enhance the power of skip connections by allowing selective skipping via the use of "gates".

We will see this shortly.

```
In [3]: print("Done")
```

```
Done
```