

Introduction to Machine Learning

Day 11: Backpropagation, Activations, and Initialization

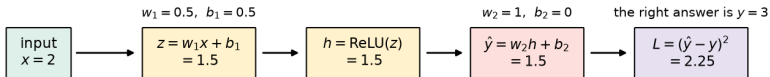
Joanna Bieri DATA301

Get out your handwritten notes!

Mini Review - What Does `loss.backward()` Do?

REVIEW

Forward pass: left to right, compute and save every value



Backward pass: right to left, the chain rule, one step at a time

$$\begin{array}{ccccccc} \frac{\partial L}{\partial w_1} = -3 \cdot x = -6 & \longleftarrow & \frac{\partial L}{\partial z} = -3 \cdot 1 = -3 & \longleftarrow & \frac{\partial L}{\partial w_2} = -3 \cdot h = -4.5 & \longleftarrow & \frac{\partial L}{\partial y} = 2(\hat{y} - y) = -3 \\ & & \frac{\partial L}{\partial b_1} = -3 & & \frac{\partial L}{\partial h} = -3 \cdot w_2 = -3 & & \end{array}$$

That is all `loss.backward()` does: the chain rule, one step at a time, starting at the loss and going backward through the network. This is called **backpropagation**.

Mini Review - A Deep Network That Does Not Learn

REVIEW

10 hidden layers of 50 neurons each, with the **sigmoid** activation, trained the same way as last class.

```
epoch 1    valid RMSE: 1.1512
epoch 5    valid RMSE: 1.1535
epoch 10   valid RMSE: 1.1525
```

validation RMSE if we always predict the average: 1.1503

Mini Review - Look at the Gradients

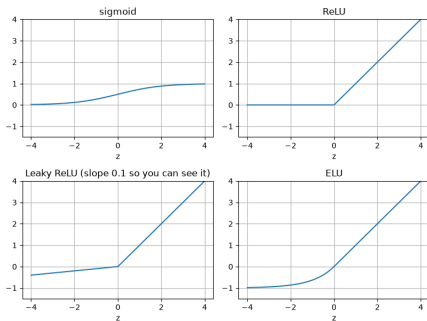
REVIEW

```
layer 1    average gradient size: 0.0
layer 2    average gradient size: 0.0
layer 3    average gradient size: 3e-08
...
layer 9    average gradient size: 0.00544868
layer 10   average gradient size: 0.03911236
layer 11   average gradient size: 2.35417581
```

A weight with a gradient of 0 does not change, no matter how many steps we take. This is called the **vanishing gradients problem**.

Mini Review - Better Activation Functions

REVIEW



- **ReLU**, **Leaky ReLU** (`nn.LeakyReLU()`) and **ELU** (`nn.ELU()`).
- **ReLU is the default choice**, and the others are things to try if ReLU is not working.

Mini Review - How Big Is the Signal?

REVIEW

The standard deviation of the numbers coming out of each hidden layer, for three sets of starting weights:

	after layer 1	after layer 10
default	0.33412	0.05001
TOO SMALL (std 0.01)	0.01577	0.0
TOO BIG (std 1)	1.57743	2827710.5

The starting weights need to be just the right size.

Mini Review - Kaiming Initialization

REVIEW

```
for layer in kaiming_model:
    if isinstance(layer, nn.Linear):
        nn.init.kaiming_normal_(
            layer.weight,
            nonlinearity="relu")
        nn.init.zeros_(layer.bias)
```

- PyTorch calls it **Kaiming**, the reading calls it **He**.
- The weights are random numbers with standard deviation $\sqrt{2/\text{number of inputs}}$.
- Now the signal stays between about 0.5 and 0.9 all the way through the 10 hidden layers.

Mini Review - Training the Deep Network

REVIEW

10 hidden layers, validation RMSE after 20 epochs

activation	starting weights	learning rate	RMSE
sigmoid	default	0.01	1.15
ReLU	default	0.01	0.5754
ELU	default	0.01	0.5988
ReLU	Kaiming	0.001	0.6076

With the sigmoid a deep network does not learn at all. ReLU, ELU and Kaiming initialization all fix that.

Backpropagation

ASK

In your own words,
describe what `loss.backward()` does.

Why does it go backward?

Backpropagation

ANSWER

Describe what `loss.backward()` does. Why does it go backward?

- **Backpropagation** finds the derivative of the loss for every weight.
- It is the chain rule: start at the loss and go back one layer at a time, multiplying by one more derivative at each step.
- It goes backward because each derivative needs the one after it, and it reuses the values saved in the forward pass.

Vanishing Gradients

ASK

What is a vanishing gradient?

Vanishing Gradients

ANSWER

What is a vanishing gradient?

- Going back through the network, the gradient shrinks at each layer.
- By the first layers it is almost 0, so those weights barely change.
- The first layers are the ones that look at the features, so the network cannot learn from them.

Vanishing Gradients

ASK

The derivative of the sigmoid
is never bigger than 0.25.

Why is that a problem with 10 layers
and not with 1?

Hint. Every step back multiplies by one more derivative. What is 0.25^{10} ?

Vanishing Gradients

ANSWER

The derivative of the sigmoid is never bigger than 0.25. Why is that a problem with 10 layers and not with 1?

- Going back through each sigmoid layer multiplies the gradient by 0.25 or less.
- One layer: times 0.25. Ten layers: $0.25^{10} \approx 0.000001$.
- The first layers get a gradient that rounds to 0.

Better Activation Functions

ASK

What is a dead ReLU neuron?

How does Leaky ReLU

keep that from happening?

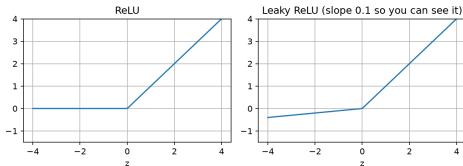
Hint. What does ReLU output for a negative z ? What is its derivative there?

Better Activation Functions

ANSWER

What is a dead ReLU neuron? How does Leaky ReLU prevent it?

- A neuron whose z is negative for every row. It outputs 0 and its gradient is 0, so it can never recover.
- Leaky ReLU gives $0.01z$ for a negative z , a small slope instead of a flat 0.
- A small gradient gets through, so the neuron can recover.

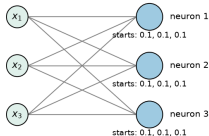


Where Should the Weights Start?

ASK

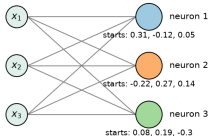
Why can we not start every weight
in a layer at the same number?

Every weight starts at 0.1



same start, so the same output,
the same gradient and the same step.
After training they are STILL identical.
3 neurons that act like 1.

Random starting weights



different starts, so different outputs
and different gradients.
Each neuron can learn
something different.

Where Should the Weights Start?

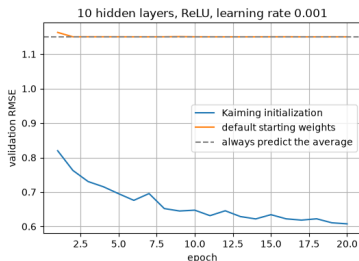
ANSWER

Why can we not start every weight in a layer at the same number?

- Every neuron computes the same output, gets the same gradient and takes the same step.
- The 3 hidden neurons stay identical forever. We paid for 3 neurons and got 1.
- Random starting weights **break the symmetry**, so each neuron can learn something different.

Training the Deep Network

ASK



10 hidden layers of 50 neurons, ReLU, learning rate 0.001. The only difference is the starting weights. Why does one curve never leave the dashed line?

Training the Deep Network

ANSWER

Only the starting weights differ. Why does the default curve never leave the dashed line?

- The default starting weights are small (standard deviation 0.08, Kaiming asks for 0.2).
- The signal shrinks from 0.33 to 0.05 going forward, and the gradient in layer 1 is about 0.000007.
- A learning rate of 0.001 times a gradient that small is too small a step to change the weights.

Questions?

Questions?