Intuition for the Regularization parameter lambda: λ

$Q)$ How does regularization prevent overfitting? If we increase the valve of λ to be very high, the value of W will become $Zero$. can someone explain this math?

$A1)$

> What L2 regularization does is add another term to the cost which is proportional to $\lambda * \sum ||W^{[l]}||_2$. The Frobenius (L_2) norm of a matrix is the sum of the squares of all the elements of the matrix. If you include that as one of the terms of cost and then try to minimize the cost, it results in the absolute values of the weights being reduced.
> Each iteration of gradient descent is trying to make the overall cost lower, which includes both the base cost and the regularization term.

$A2)$ $BetterAnswer$

> The optimisation algorithm is trying to minimise the cost function but L_2 regularisation has added on the squared norm of W onto to end of the cost function.
>
> The squared norm (strictly squared Frobenius norm) is the sum of the square of all of the elements of W. We then also sum this over all the layers in the network and multiply by $\lambda/(2 * m)$.
>
> So if the elements of W are large, this is going to get pretty big very quickly - especially if λ is big. So the easiest way for the optimiser to reduce the cost is to make the elements of W small and the larger the λ, the smaller it's going to want to make the elements of W - eventually making them very close to zero for large vales of λ.
>
> As mentioned in the video, L_2 regularisation is also sometimes called weight decay because each time through the loop, W is getting reduced by $(\alpha * \lambda/m) * W$ - so the larger the value of $lambda$, the quicker the weights will decrease towards 0 during the optimisation process.
>
> This has the effect of effectively switching off some of the units because the weights are so close to 0 - thus making the network simpler (with less units) and hence reducing the chance of it overfitting.
>
> You want to minimize this expression: $min(L + \lambda \sum_i w_i^2)$ (where L is the loss like mean squared error for regression or cross entropy for classification.)
>
> If λ is small then the optimizer will focus in minimizing L. If λ is big, then the optimizer will focus on the right-hand side of the expression, which will make him reduce the magnitude of each w.