

Define and .

**Forward Feeding**

where we adopt the sigmoid function

**Back Propagation**

We use an improper gradient notation

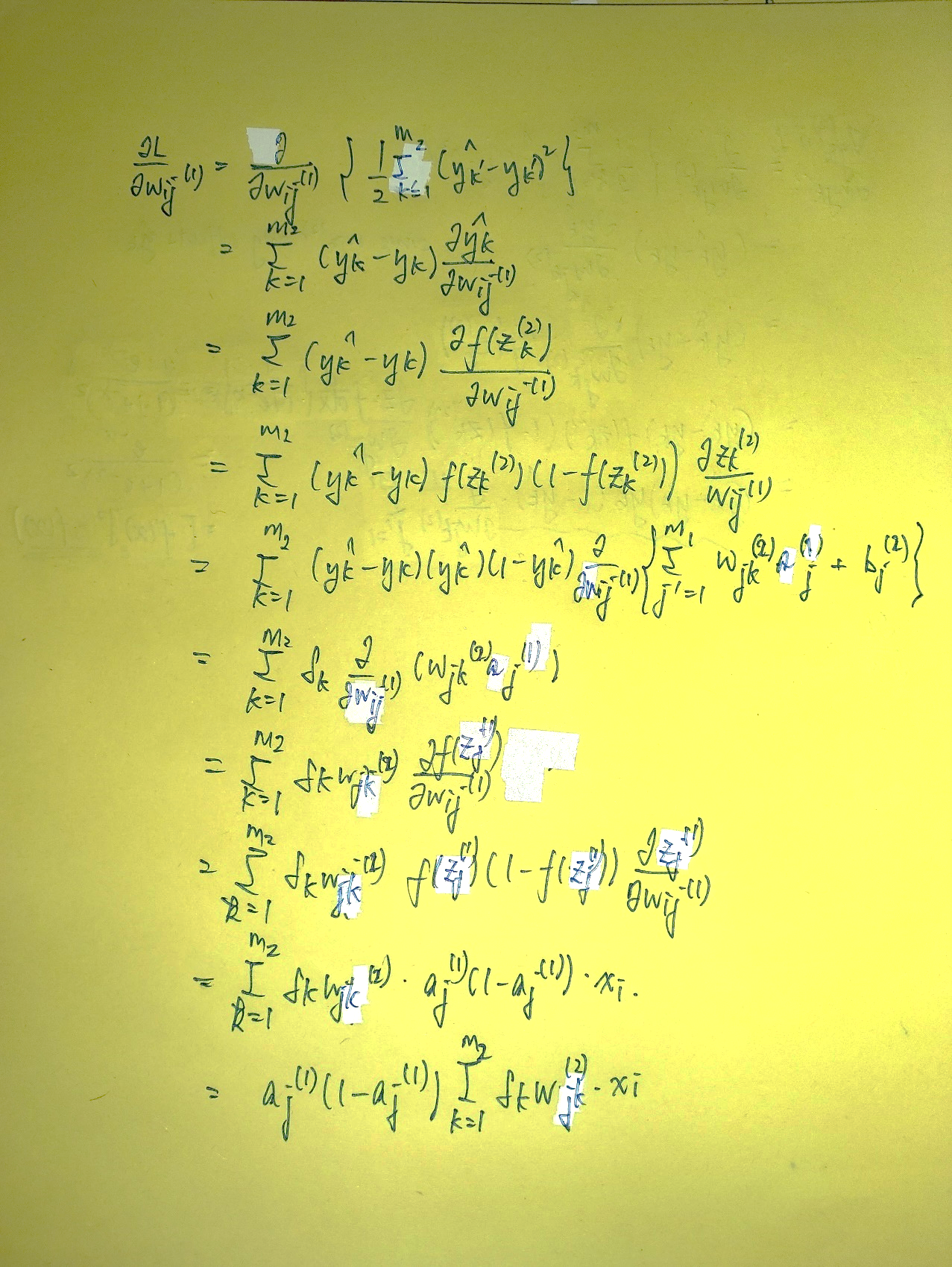
MSE Loss Function:

For output layer (layer 2),

For hidden layer (layer 1),

**Derivation**

**

**

*Q1.5*

[*https://www.cs.cmu.edu/~aarti/Class/10315\_Spring22/315S22\_Rec4.pdf*](https://www.cs.cmu.edu/~aarti/Class/10315_Spring22/315S22_Rec4.pdf)

*is a function of ()*

*In matrix form, (Denominator-layout notation)*

*Q2.3*

Derivative of cross-entropy(softmax(x)):

Derivatives for output layer:

Dimensions: .

Derivatives for hidden layers:

Note that since our network only has two layers, .

*Q5.1.1*

1

*Q1.6*

*Q6*

[*https://www.sjsu.edu/faculty/guangliang.chen/Math253S20/lec5svd.pdf*](https://www.sjsu.edu/faculty/guangliang.chen/Math253S20/lec5svd.pdf)

[*https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0d2a4d8*](https://towardsdatascience.com/pca-and-svd-explained-with-numpy-5d13b0d2a4d8)

[*https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca*](https://stats.stackexchange.com/questions/134282/relationship-between-svd-and-pca-how-to-use-svd-to-perform-pca)

*For square matrices, the spectral decomposition is*

*We define a similar decomposition for non-square matrices, the SVD, as*

*In matrix form,*

*PCA is performed by taking the first K* *terms of the sum of the covariance matrix to approximate the matrix X.*

*This can be done using the svds(X, K) function.*

Let’s test this on a neural network with one hidden layer using a sigmoid activation and output layer function.

There are three layers, the input layer (l=0), the hidden layer (l=1) and the output layer (l=2).

With zero-initialization, all weights and biases are 0.

For forward propagation,

The network outputs and regardless of the input .

The gradients are

where and .

We see that for our zero-initialized network, and the output layer weights are not updated.

Furthermore, and the hidden layer weights are not updated.

If we generalize this result to a network with any activation function, then

where and are the outputs before applying the activation function. We see these two are also 0.