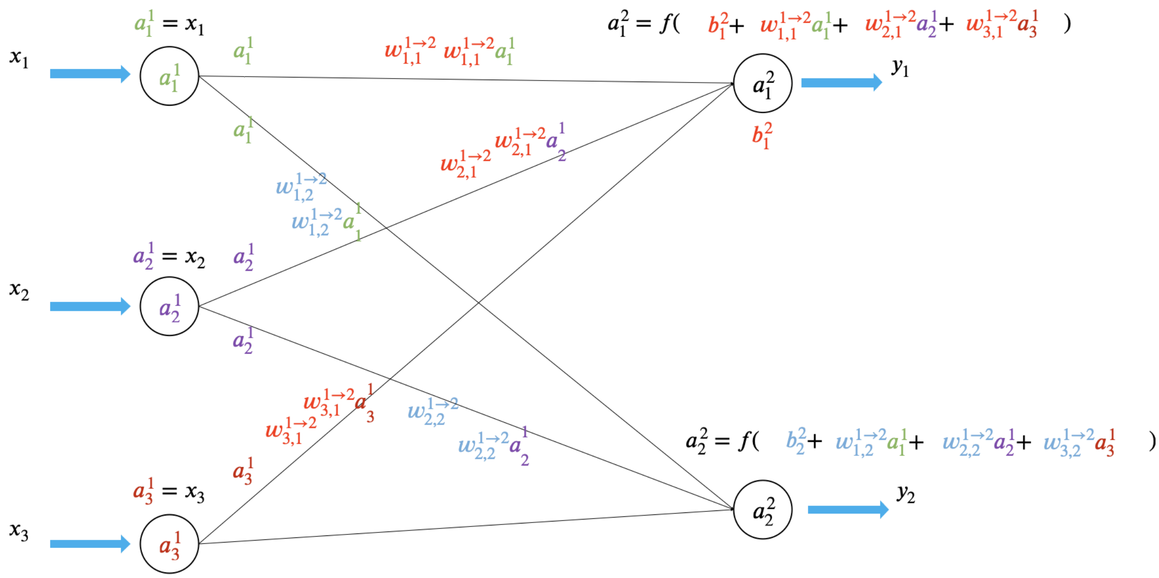
CAS CS 440
Lec 24

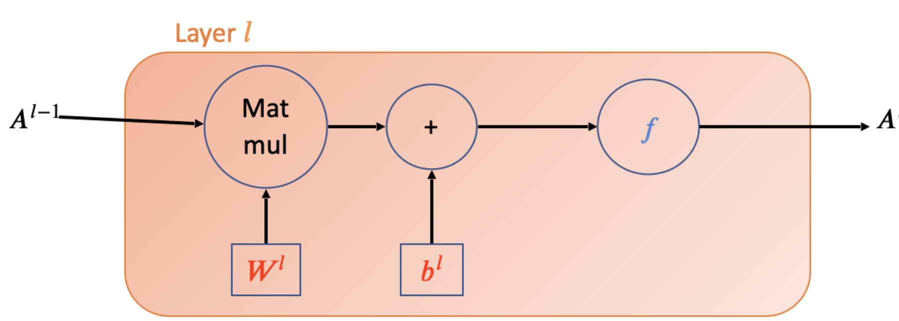
Supervised Learning VI – Neural Networks (cont.)

1. Neural Networks: Forward Propagation
    a. 

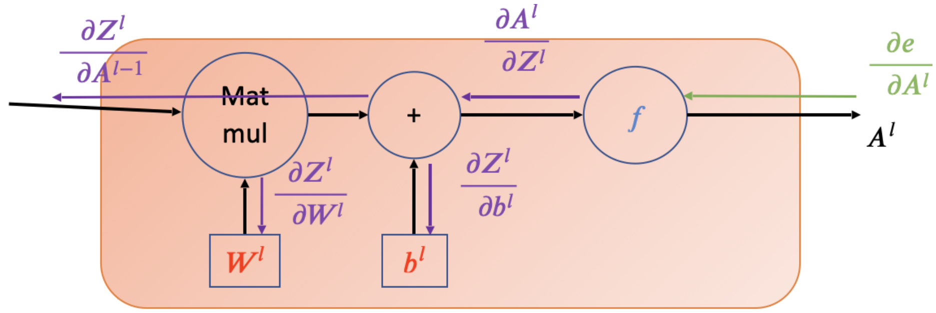
2. NN as Computation Graph
    a. Given a “batch” of examples X and the ground truth output Ygf
        i. We know forward equations:

$$Z^l = A^l W^{l-1\to l} + b^l$$

$$A^l = f(Z^l)$$

        ii. Let’s view a layer as a computation graph
            1. Layer has no idea where it is in the NN
                a. That’s ok! Chain rule to the rescue!
            2. 

3. NN Computation Graph Backprop
   a. Each forward edge gets a backwards edge for chain rule!
      i. We know these equations!



      ii.

$$\frac{\partial A^l}{\partial Z^l} = f'(Z^l)$$

$$\frac{\partial Z^l}{\partial b^l} = \mathbf{1}^T$$

$$\frac{\partial Z^l}{\partial W^l} = \left(A^{l-1}\right)^T$$

      iii. $$\frac{\partial Z^l}{\partial A^{l-1}} = \left(W^l\right)^T$$

   b. Combine pieces of chain rule to get the derivatives we want

$$\frac{\partial e}{\partial Z^l} = f'(Z^l)\frac{\partial e}{\partial A^l} \qquad \frac{\partial e}{\partial b^l} = \mathbf{1}^T\frac{\partial e}{\partial Z^l} \qquad \frac{\partial e}{\partial W^l} = \frac{\partial e}{\partial Z^l}\left(A^{l-1}\right)^T \qquad \frac{\partial e}{\partial A^{l-1}} = \left(W^l\right)^T\frac{\partial e}{\partial Z^l}$$

   c. If we assume we have a “line graph”

   d. procedure forward(Matrix X) -> Matrix:

```
for(layer in layers):
    X = layer.forward(X)
return X // now final predictions
```

   e.

```
procedure backwards(Matrix X, Matrix de_dA) -> Matrix:
    // need to do a forward pass and cache the intermediary values
    Stack[Matrix] cache = [X]
    for(layer in layers):
        X = layer.forward(X)
        if layer is not last layer then: cache.push(X)

    // now start at the last layer and walk backwards to the front
    for(layer in reversed(layers)):
        de_dA = layer.backwards(cache.pop(), de_dA)
    return de_dA // now de_dX at this point
```