MLP Backpropogation math

```
def forward(self, x):
  x1 = x @ w1.T
  x2 = sigmoid(x1)
  x3 = x2 @ w2.T
  x4 = softmax(x3)
  return x4
```

Basic matrix function derivatives

1. Notice that, for scaler function derivatives, we can directly use chain rules to get the target variable derivative. However, for scaler-matrix function derivative, if we use chain rules, we may face to very complicated matrix-matrix function derivative. To avoid such dilemma, we will use the total derivative to get the final results.
2. Some basic rules:

$$\begin{aligned}
&d(X \pm Y) = dX \pm dY, d(XY) = dXY + XdY \\
&dX \odot Y = dX \odot Y + X \odot dY \\
&d(\sigma(X)) = \sigma'(X) \odot dX \\
&tr(AB) = tr(BA) \\
&tr(A^T(B \odot C)) = tr((A \odot B)^T C)\$, \$A, B, C \in \mathbb{R}^{n \times n} \\
&df = tr(\frac{\partial f}{\partial X}^T dX)
\end{aligned}$$

3. We will use formula (6) as the very basic beginning for getting our derivatives:

Backpropogation starts at loss function

$$\begin{gathered}
l = -y \, \text{logsoftmax}(\sigma(XW_1^T)W_2^T)^T \\
X \in \mathbb{R}^{1 \times dim1}, W_1 \in \mathbb{R}^{dim2 \times dim1}, W_2 \in \mathbb{R}^{ncl \times dim2} \\
a_1 = XW_1^T, h1 = \sigma(a_1), a_2 = h_1 W_2^T \\
\text{Find: } \frac{\partial l}{\partial W_1} \text{and} \frac{\partial l}{\partial W_2}
\end{gathered}$$

In first way, we get to $$\frac{\partial l}{\partial a_2}$$, in a very traditional way:

$$l = -y \log \text{softmax}(a_2)^T$$
$$l = -y(\log(\exp(a_2)) - \mathbf{1}_{ncl \times ncl} \log(\exp(a_2)\mathbf{1}_{ncl \times 1}))^T$$
$$l = -ya_2^T + \log(\mathbf{1}_{ncl \times 1}^T \exp(a_2^T))$$
$$dl = -ydW_2h_1^T + \frac{\mathbf{1}_{ncl \times 1}^T(\exp(W_2h_1^T) \odot (dW_2h_1^T))}{\mathbf{1}_{ncl \times 1}^T \exp(W_2h_1^T)}$$
$$dl = -ydW2h_1^T + \frac{\exp(h_1W_2^T)dW_2h_1^T}{\mathbf{1}_{ncl \times 1}^T \exp(W_2h_1^T)}$$
$$dl = tr(h_1^T(-y) + h_1^T \frac{\exp(h_1W_2^T)}{\mathbf{1}_{ncl \times 1}^T \exp(W_2h_1^T)} dW_2)$$
$$dl = tr(-h_1^Ty + h_1^T \text{softmax}(h_1W_2^T)dW_2)$$
$$\frac{\partial l}{\partial W_2} = (-y^T + \text{softmax}(W_2h_1^T))h_1$$

Or, we can have:

$$\frac{\partial l}{\partial a_2} = \text{softmax}(a_2) - y \quad (*1)$$
$$dl = tr(\frac{\partial l}{\partial a_2}^T da_2) = tr(\frac{\partial l}{\partial a_2}^T dh_1W_2^T + \frac{\partial l}{\partial a_2}^T h_1dW_2^T)$$
$$dl = tr((\frac{\partial l}{\partial a_2}W_2)^T dh_1) + tr((\frac{\partial l}{\partial a_2}^T h_1)^T dW_2)$$
$$\frac{\partial l}{\partial W_2} = \frac{\partial l}{\partial a_2}^T h_1 = (-y^T + \text{softmax}(W_2h_1^T))h_1 \quad (*2)$$

The results are the same. Obviously, the later is much simpler.

On top of that, we need to get further results:

$$dl_2 = tr((\frac{\partial l}{\partial a_2}W_2)^T dh_1) \text{ ignore } W_2 \text{ part for further formula}$$
$$\frac{\partial l}{\partial h_1} = \frac{\partial l}{\partial a_2}W_2 \quad (*3)$$
$$dl_2 = tr((\frac{\partial l}{\partial h_1})^T d\sigma(a_1))$$
$$dl_2 = tr((\frac{\partial l}{\partial h_1})^T \sigma'(a_1) \odot da_1)$$
$$dl_2 = tr((\frac{\partial l}{\partial h_1} \odot \sigma'(a_1))^T da_1)$$
$$\frac{\partial l}{\partial a_1} = \frac{\partial l}{\partial h_1} \odot \sigma'(a_1) = \frac{\partial l}{\partial a_2}W_2 \odot \sigma'(a_1) \quad (*4)$$
$$dl_2 = tr((\frac{\partial l}{\partial a_1})^T da_1) = tr((\frac{\partial l}{\partial a_1})^T XdW_1^T)$$
$$dl_2 = tr(((\frac{\partial l}{\partial a_1})^T X)^T dW_1)$$
$$\frac{\partial l}{\partial W_1} = \frac{\partial l}{\partial a_1}^T X \quad (*5)$$

Matrix Function Derivatives Methods:

1.We need a universal formula for matrix-matrix function derivatives:

$$\text{vec}(dF) = \frac{\partial F}{\partial X}^T \text{vec}(dX)$$

$$2. Composition of matrix - matrix function derivatives:$$

$$\text{vec}(dF) = \frac{\partial F}{\partial Y}^T \text{vec}(dY) = \frac{\partial F}{\partial Y}^T \frac{\partial Y}{\partial X}^T \text{vec}(dX)$$

And we could get use the chain rules for the complicated gradient computations

Properties:

$\text{vec}(A + B) = \text{vec}(A) + \text{vec}(B)$

$\text{vec}(AXB) = (B^T \otimes A)\text{vec}(X)$

$\text{vec}(A^T) = K_{mn}\text{vec}(A), A \in \mathbb{R}^{m\times n}, K_{mn} \in \mathbb{R}^{mn\times mn}, \quad K_{mn}$ is commutation matrix

$\text{vec}(A \odot X) = \text{dial}(A)\text{vec}(X), \quad \text{diag}(A) \in \mathbb{R}^{mn\times mn}$ is a diagonal use elements in A, oredered by columns

$A \otimes B)^T = A^T \otimes B^T$

$\text{vec}(ab^T) = b \otimes a$

$(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$

$K_{mn} = K_{nm}^T, K_{mn}K_{nm} = I$

$K_{pm}(A \otimes B)K_{nq} = B \otimes A, A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{p\times q}$