Section 2: Loss Function.

- $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ (L2 loss)

  Variation: - Half of MSE: $\frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

  - Root MSE: $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$

- $MAE = \frac{1}{2}\sum_{i=1}^{n}|y_i - \hat{y}_i|$ (L1 loss)

L2 loss is more sensitive to outliers than L1 loss, L1 loss is more robust and generally not affected by outliers.
Ex: If error is 10, MAE give 10 and MSE would give 100

- Huber Loss (combine L1 and L2)

$$L_\gamma(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for } |y - f(x)| \leq \gamma \\ \gamma|y - f(x)| - \frac{1}{2}\gamma^2 & \text{otherwise} \end{cases}$$

- Binary Cross Entropy:

$$BCE = -\frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{n}\left[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)\right]$$

y: actual label (0 or 1) — c: number of class
p: predicted proba for class — n: number of samples.

- Cross Entropy:

$$CE = -\frac{1}{n}\sum_{j=1}^{n}\sum_{c=1}^{n} y_i \log(\hat{y}_i)$$

c: number of classes, y_i: actual label, $\hat{y}_i$: predicted label (can be proba)

This loss only calculated for correct prediction (0 otherwise)
It penalizes probabilities of correct class only

- Softmax function: $\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$.

- KL divergence loss: difference between the entropy and cross entropy loss

$$KL(y||\hat{y}) = \sum_i y_i \log \frac{1}{\hat{y}_i} - \sum_i y_i \log \frac{1}{y_i} = \boxed{\sum_i y_i \log \frac{y_i}{\hat{y}_i}}$$

Major use: Variational Autoencoders.

In image classification, we use one-hot encoding for our labels.
Therefore, y_i is the actual label, it equals 1 → $\log(1) = 0$
when y_i is not, x also be cancelled out.
So KL divergence = cross entropy in image classification tasks

- Contrastive loss: distance-based loss function

$$\mathcal{L}_{contra} = yD^2 + (1-y)\cdot \max(0, m-D)^2$$

, here D is the Euclidean distance and y is the label

$$d = \sqrt{[O(x_1) - O(x_2)]^2}$$

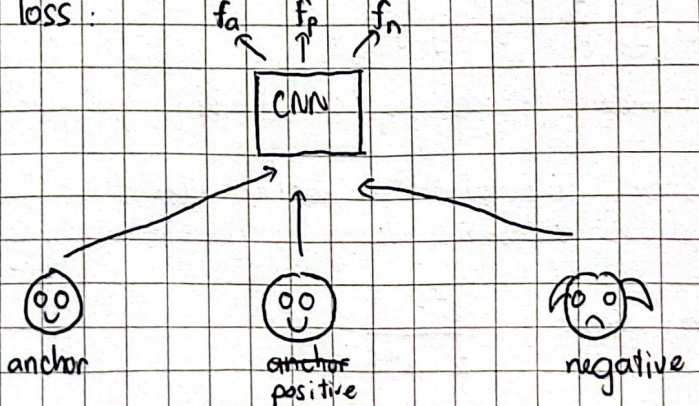
+m or margin is used for confidence. If two images in a pair are dissimilar, their distance should be at least margin

- Hinge loss: a marginal loss, usually used for SVMs, used when labels are [-1, 1].

It penalizes not only wrong predictions, but correct prediction which are not confident enough.

$$\mathcal{L}_{Hinge} = \sum \max(0, 1 - y\hat{y})$$

- Triplet ranking loss:



$$d(x,y) = \|x-y\|_2, \quad \mathcal{L}_{Triplet}(r_a, r_p, r_n) = \max(m + d(r_a, r_p) - d(r_a, r_n), 0)$$

There are 3 situation

+) $d(r_a, r_n) > d(r_a, r_p) + m$

$\Rightarrow \max(0, \text{negative}) = 0 \rightarrow$ no loss $\rightarrow$ Perfect.

+) $d(r_a, r_p) > d(r_a, r_n)$

$\Rightarrow \max(0, m + \text{positive}) = m + \text{positive} \rightarrow$ Loss is greater than m.

$\rightarrow$ Model tend to minimize the loss, bad.

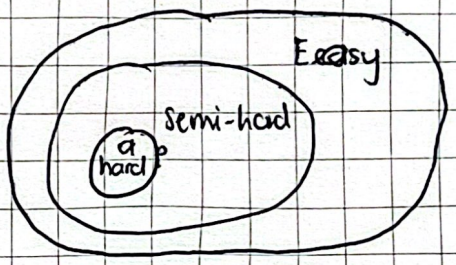
+) $d(r_a, r_p) < d(r_a, r_n) < d(r_a, r_p) + m$

For short, . $d(r_a, r_n) > d(r_a, r_p) + m$ is good position, no loss, or loss is

. $d(r_a, r_p) > d(r_a, r_n)$, bad! , must count this in loss

. $d(r_a, r_p) < d(r_a, r_n) < d(r_a, r_p) + m$, ok! in this case (semiha

they are at the margin



How should we use choose the triplets?

Online triplet mining : triplets are defined for every batch during the training

For example, training w/ easy triplets should be avoid, no loss account.

FaceNet(gg) use Semi-hard triplet mining, ~~choose tri~~