- To explain the clip feature extraction to loss computation with a toy example, we will cover each key stage.

1) image & text feature extraction.
2) similarity computation.
3) loss computation.

- lets use a sample batch of 3 image-text pair for clarity.

1 Toy Example set up.

- We have a batch of 3 image-text pairs.
  - I_1 matches T_1
  - I_2 matches T_2
  - I_3 matches T_3

step 1 Extract embeddings for images:

$$I_e = \begin{bmatrix} 0.8 & 0.2 & 0.1 \\ 0.1 & 0.9 & 0.2 \\ 0.3 & 0.1 & 0.9 \end{bmatrix} \begin{array}{l} I_1 \\ I_2 \\ I_3 \end{array}$$

- each row corresponds to the embedding of an image. (dimensionality = 3)

2. Text Encoding: Extract embedding of text

$$T_e = \begin{array}{c} \begin{bmatrix} 0.9 & 0.1 & 0.0 \\ 0.2 & 0.8 & 0.0 \\ 0.1 & 0.2 & 0.9 \end{bmatrix} \end{array} \begin{array}{l} T_1 \\ T_2 \\ T_3 \end{array}$$

each row is corresponds to a text ~~embedding~~ description $(d=3)$

$\boxed{\text{STEP 2}}$

we compute the similarity b/w all image_text pairs (not just aligned pairs). cosine similarity is given by

$$\text{similarity}(I_i, T_j) = \frac{I_i \cdot T_j^{T}}{\|I_i\| \cdot \|T_j\|}$$

just to make the dot product b/w the image & text

For simplicity, assume embeddings are normalized (i.e $\|I_i\| = \|T_i\| = 1$). Then similarity is just a dot product

Image →

$$\text{logits} = I_e \cdot T_i^{T} = \begin{array}{c} I_1 \\ I_2 \\ I_3 \end{array} \begin{bmatrix} 0.8 & 0.2 & 0.1 \\ 0.1 & 0.9 & 0.2 \\ 0.3 & 0.1 & 0.9 \end{bmatrix}_{3\times3} \begin{array}{c} \begin{array}{ccc} T_1 & T_2 & T_3 \end{array} \\ \begin{bmatrix} 0.9 & 0.2 & 0 \\ 0.1 & 0.8 & 0.2 \\ 0.0 & 0.0 & 0.9 \end{bmatrix}_{3\times3} \end{array}$$

Text

why?

$e^{(T)}$

why?

logit means raw output $z = wx + b$ before apply softmax/sigmoid

$$= \begin{vmatrix} 0.74 & 0.26 & 0.20 \\ 0.29 & 0.74 & 0.26 \\ 0.30 & 0.20 & 0.85 \end{vmatrix}$$

each row corresponds to an image, and each column corresponds to a text

$\boxed{\text{Step 3 apply softmax}}$ ⇒ convert logits into probabilities

To calculate probability, apply softmax to each row (image-to-text direction) & each ~~column~~ column (text-to-image direction)

$\boxed{\text{Image-to-text } \cancel{\text{direction}} \text{ softmax}}$

For row I_1 (0.74, 0.26, 0.20)

$$\text{Softmax}(I_1) = \left(\frac{e^{0.74}}{e^{0.74}+e^{0.26}+e^{0.20}}, \frac{e^{0.26}}{e^{0.7}+e^{0.26}+e^{0.20}}, \frac{e^{0.20}}{e^{0.7}+e^{0.26}+e^{0.20}} \right)$$

$\text{Softmax}(I_1) = [0.492, 0.260, 0.248]$

~~$\text{Softmax}(I_2) = [0.258, 0.486, 0.256]$~~

~~$\text{Softmax}(I_2) = [0.250, 0.249, 0.501]$~~

$\text{Softmax}(I_2) = [0.26, 0.492, 0.248]$

$\text{Softmax}(I_3) = [0.265, 0.241, 0.494]$

$\boxed{\text{Text to Image softmax}}$

$\text{Softmax}(t_1) = [0.46, 0.264, 0.271]$

$\text{Softmax}(t_2) = [0.258, 0.486, 0.256]$

$\text{Softmax}(t_3) = [0.250, 0.249, 0.501]$

step 4 cross entropy

The ground Truth label [0, 1, 2]

Image-to-text loss (Loss1):

For I_1 (label=0) [1, 0, 0]

$$loss(I_1) = -\log(0.492) = 0.709$$

for I_2 (label = 1) [0, 1, 0]

$$loss(I_2) = -\log(0.492) = 0.709$$

for I_3 (label=2) [0, 0, 1]

$$loss(I_3) = -\log(0.494) = 0.705$$

np.exp(t)

$$\text{Average loss1} = \frac{0.709 + 0.709 + 0.705}{3} = 0.708$$

cross entropy

$$\boxed{loss = -\sum_{i=1}^{n} y_i \cdot \log \hat{y}_i}$$

Text-to-Image loss (Loss 2):

For T_1 (label=0): [1 0 0]

$$loss(T_1) = -\log(0.465) = 0.766$$

For T_2 (label=1) [0 1 0]

$$loss(T_2) = -\log(0.486) = 0.721$$

For T_3 (label=2) [0 0 1]

$$loss(T_3) = -\log(0.51) = 0.691$$

$$loss_2 = \frac{0.766 + 0.721 + 0.691}{3}$$

$$\text{Avg. } loss_2 = 0.726$$

step 5

$$loss = \frac{Loss1 + loss2}{2} = \frac{0.708 + 0.726}{2}$$

$$= 0.717$$

Summary:

- Feature extraction: creates embeddings / numerics values for image & text
- cosine similarity: b/w embeddings from logits matrix.
- Softmax: convert logits into probabilities.
- cross entropy: penalizes misalignments, encouraging matching image-text pairs

Temperature Scaling

Image 1: A clear picture of a Cat (Label: Cat)

Image 2: A clear picture of a Dog (Label: Dog)

Image 3: A blurry picture of a Cat (Label: Cat)

Image 4: A blurry picture of a Dog (Label: Dog)

RAW Similarity Score

		cat T_1	dog T_2
clear cat	I_1	0.9	0.1
clear Dog	I_2	0.2	0.8
blurry Cat	I_3	0.6	0.4
blurry Dog	I_4	0.3	0.5

Adjusting CONFIDENCE LEVEL

(A) ~~XX~~

(as the model is very confident in its prediction & may not learn effectively from other, more ambiguous images.)

w/o temperature Scaling

we apply softmax and & we get

		cat T	dog T_2
clear cat	I_1	(overconfident) 0.92	0.1
clear Dog	I_2	0.2	(0.8) (very confident)
blurry Cat	I_3	0.6	0.4
blurry Dog	I_4	0.3	0.5

Applying temperature scaling

Now, let's apply temperature scaling $T=2$

Similary$(I, t)/2$

	T_1	T_2
I_1	0.45	0.05
I_2	0.1	0.4
I_3	0.3	0.2
I_4	0.15	0.25

- New probabilities after scaling

original RAW similarity (logits)

original			T_1	T_2
0.9	0.1	I_1	0.6	0.4
0.2	0.8	I_2	0.425	0.575
0.6	0.4	I_3	0.545	0.425
0.3	0.8	I_4	0.475	0.525

By applying a higher temperature (like $T=2$) the model's confidence is tempered, resulting in more balanced probability. By adjusting a probability of 0.95 for cat + 0.60 indicating that there is still some uncertainty about classification

⇒ uncertain → increase loss → gradient high
Weights update to learn complex features.

⇒ more robust feature learned

⇒ clear & unclear Image. soft cases feature learned

⇒ learn feature in soft clear & unclear Image. → balance learning.