

Contrastive loss and the InfoNCE gradient

Ten problems on contrastive learning: InfoNCE as a softmax cross-entropy over similarities, the gradient of cosine similarity, the gradient with respect to the anchor and to the positive and negative candidates, what the temperature does to the weighting of negatives, the symmetric CLIP-style batch loss and its logit-matrix gradient, scale invariance of the normalised loss, the gradient with respect to a learned temperature, the matrix form through the normalisation, and why unnormalised dot-product logits are not scale invariant, with worked solutions and the mistakes that drop the projection, the $1/\tau$, the column loss or the sign of the temperature gradient.

BEFORE YOU START

A contrastive loss asks an encoder to place an anchor close to its positive and far from a set of negatives, and InfoNCE, the loss behind SimCLR, MoCo and CLIP, does it with a softmax over similarities: it is cross-entropy with the positive as the label. So its gradient with respect to the logits is the softmax page's $p - y$, and everything else on this page is the chain rule through two things that page did not have: a cosine similarity, which normalises both vectors, and a temperature. These ten problems compute the gradient of the cosine, push $p - y$ through it to the anchor and to each candidate, read off what the temperature does to the weights on hard and easy negatives, do the symmetric two-direction batch loss used by CLIP, differentiate with respect to a learned temperature, and show why the normalisation is what stops the encoder from cheating by scaling. The five mistakes each leave a working training run: a cosine gradient that forgets the norm, a dropped $1/\tau$, a one-directional batch loss, a normalisation backward without its projection, and a temperature gradient with the wrong sign.

- Vectors are columns and gradients have the shape of their variable, as on the earlier pages. $\hat{u} = u/\|u\|$ is the unit vector along u , and I is the identity.
- The anchor is $u \in \mathbb{R}^d$ and the candidates are $v_1, \dots, v_K \in \mathbb{R}^d$; candidate 1 is the positive and the rest are negatives. $y = e_1$ is the one-hot label.
- Cosine similarity is $c(u, v) = \frac{u^\top v}{\|u\|\|v\|} = \hat{u}^\top \hat{v}$, and $c_j = c(u, v_j)$.
- The temperature $\tau > 0$ turns similarities into logits $z_j = c_j/\tau$; $p = \text{softmax}(z)$, so $p_j = e^{z_j} / \sum_k e^{z_k}$, and $\sum_j p_j = 1$.
- InfoNCE for one anchor is $\ell = -\log p_1 = -z_1 + \log \sum_j e^{z_j}$. The softmax page showed $\nabla_z(-\log p_1) = p - y$ (its Problem 5) and that $\text{softmax}(z/T)$ sharpens as $T \rightarrow 0$ (its Problem 9).

- The Jacobians page's Problem 9 gives $\partial \hat{u} / \partial u = \frac{1}{\|u\|} (I - \hat{u} \hat{u}^\top)$, and the matrix-calculus page gives $\nabla_u \|u\| = \hat{u}$ and $\nabla_u (a^\top u) = a$.
- Batch form: N pairs (u_i, v_i) , with $\hat{U}$ and $\hat{V}$ the $N \times d$ matrices of unit rows and $Z = \hat{U} \hat{V}^\top / \tau$ the $N \times N$ logit matrix, $Z_{ij} = c(u_i, v_j) / \tau$. P_r is the row-wise softmax of Z and P_c the column-wise softmax.

Builds on: [The softmax Jacobian](#), [Jacobians and the chain rule](#)

Problems

1. Show that $\ell = -z_1 + \log \sum_j e^{z_j}$ equals $-\log p_1$, compute $\nabla_z \ell$, and deduce $\nabla_c \ell$ for the vector c of similarities. difficulty ·

2. Compute $\nabla_u c(u, v)$ and $\nabla_v c(u, v)$. Show that $\nabla_u c$ is orthogonal to u and find its norm. difficulty ··

3. Compute $\nabla_u \ell$ for InfoNCE with cosine similarities, and simplify it to an expression in $\hat{u}$, the unit candidates $\hat{v}_j$, the weights p_j and the similarities c_j . difficulty ··

4. Compute $\nabla_{v_j} \ell$ for the positive ($j = 1$) and for a negative ($j \neq 1$), and say in which direction each candidate moves under gradient descent. difficulty ··

5. Show that the weight the loss puts on negative j relative to negative k is $p_j / p_k = e^{(c_j - c_k) / \tau}$. Describe p and $\nabla_c \ell$ as $\tau \rightarrow 0$ and as $\tau \rightarrow \infty$. difficulty ··

6. The symmetric batch loss used by CLIP is $L = \frac{1}{2N} \sum_{i=1}^N [\ell_i^{\text{row}} + \ell_i^{\text{col}}]$ with $\ell_i^{\text{row}} = -Z_{ii} + \log \sum_j e^{Z_{ij}}$ (anchor u_i against all v_j) and $\ell_i^{\text{col}} = -Z_{ii} + \log \sum_j e^{Z_{ji}}$ (anchor v_i against all u_j). Compute $\nabla_Z L$. difficulty ···

7. Show that ℓ is unchanged by $u \rightarrow \alpha u$ for any $\alpha > 0$, that $u^\top \nabla_u \ell = 0$, and that $\nabla_u \ell$ evaluated at αu is $\frac{1}{\alpha}$ times its value at u . If $u = Wx$ for a projection W and input x , compute $\nabla_W \ell$. difficulty ··

8. Compute $\partial\ell/\partial\tau$. CLIP learns the temperature through a logit scale difficulty ...
 $s = 1/\tau = e^t$ with t the trained parameter. Compute $\partial\ell/\partial t$, and say which way t moves when the positive's similarity exceeds the softmax-weighted average similarity.

9. For the symmetric batch loss of Problem 6 with $Z = \hat{U}\hat{V}^\top/\tau$, difficulty ...
compute $\nabla_{\hat{U}}L$ and $\nabla_{\hat{V}}L$ in matrix form, then $\nabla_{u_i}L$ through the normalisation $\hat{u}_i = u_i/\|u_i\|$.

10. Suppose the logits were the plain dot products $z_j = u^\top v_j/\tau$ with no difficulty ..
normalisation. Compute $\nabla_u\ell$ and $u^\top\nabla_u\ell$, and show that when the positive has the largest logit the loss can be reduced by scaling u up, without changing its direction.

Answers

1. $\ell = -\log p_1$, $\nabla_z \ell = p - y$, $\nabla_c \ell = (p - y)/\tau$
2. $\nabla_u c = \frac{\hat{v} - c \hat{u}}{\|u\|}$, $\nabla_v c = \frac{\hat{u} - c \hat{v}}{\|v\|}$; $u^\top \nabla_u c = 0$ and $\|\nabla_u c\| = \frac{\sqrt{1 - c^2}}{\|u\|}$
3. $\nabla_u \ell = \frac{1}{\tau \|u\|} \left[\sum_j p_j \hat{v}_j - \hat{v}_1 - \left(\sum_j p_j c_j - c_1 \right) \hat{u} \right]$
4. $\nabla_{v_j} \ell = \frac{p_j - y_j}{\tau \|v_j\|} (\hat{u} - c_j \hat{v}_j)$: the positive moves along $+(\hat{u} - c_1 \hat{v}_1)$, towards the anchor's direction; each negative moves along $-(\hat{u} - c_j \hat{v}_j)$, away from it, with weight p_j
5. $p_j/p_k = e^{(c_j - c_k)/\tau}$; $\tau \rightarrow 0$ gives a hinge on the hardest negative with gradient $(e_{j^*} - e_1)/\tau$; $\tau \rightarrow \infty$ gives uniform weights and a vanishing gradient
6. $\nabla_Z L = \frac{1}{2N} (P_r + P_c - 2I)$
7. $\ell(\alpha u) = \ell(u)$; $u^\top \nabla_u \ell = 0$; $\nabla_u \ell(\alpha u) = \frac{1}{\alpha} \nabla_u \ell(u)$; $\nabla_W \ell = (\nabla_u \ell) x^\top$
8. $\frac{\partial \ell}{\partial \tau} = \frac{c_1 - \sum_j p_j c_j}{\tau^2}$ and $\frac{\partial \ell}{\partial t} = \frac{\sum_j p_j c_j - c_1}{\tau}$; when $c_1 > \sum_j p_j c_j$, $\partial \ell / \partial t < 0$, so gradient descent increases t and decreases τ
9. $\nabla_{\hat{U}} L = \frac{1}{\tau} \Gamma \hat{V}$, $\nabla_{\hat{V}} L = \frac{1}{\tau} \Gamma^\top \hat{U}$, $\Gamma = \frac{1}{2N} (P_r + P_c - 2I)$; then $\nabla_{u_i} L = \frac{1}{\|u_i\|} (I - \hat{u}_i \hat{u}_i^\top) \left(\frac{1}{\tau} \Gamma \hat{V} \right)_{i,:}^\top$, and likewise for v_i with $\Gamma^\top \hat{U}$
10. $\nabla_u \ell = \frac{1}{\tau} \sum_j (p_j - y_j) v_j$, $u^\top \nabla_u \ell = \sum_j p_j z_j - z_1$; when the positive leads, $\ell(\alpha u)$ decreases in α , so the loss rewards growing $\|u\|$

Worked solutions

Problem 1

Show that $\ell = -z_1 + \log \sum_j e^{z_j}$ equals $-\log p_1$, compute $\nabla_z \ell$, and deduce $\nabla_c \ell$ for the vector c of similarities.

$$-\log p_1 = -\log \frac{e^{z_1}}{\sum_j e^{z_j}} = -z_1 + \log \sum_j e^{z_j}.$$

$$\log(a/b) = \log a - \log b \text{ and } \log e^{z_1} = z_1.$$

$$\partial \ell / \partial z_j = -\delta_{j1} + \frac{e^{z_j}}{\sum_k e^{z_k}} = p_j - y_j.$$

The first term contributes only for $j = 1$; the log-sum-exp has derivative p_j by the chain rule, $\frac{d}{dz_j} \log S = \frac{1}{S} \partial S / \partial z_j$ with $S = \sum_k e^{z_k}$. This is the softmax page's $p - y$.

$$z = c/\tau, \text{ so } \partial z_j / \partial c_j = 1/\tau \text{ and } \nabla_c \ell = \frac{1}{\tau} \nabla_z \ell.$$

Each c_j enters one logit, scaled by the constant $1/\tau$.

$$\ell = -\log p_1, \nabla_z \ell = p - y, \nabla_c \ell = (p - y)/\tau$$

Every entry of $\nabla_c \ell$ is positive for a negative ($p_j > 0$) and negative for the positive ($p_1 - 1 < 0$): gradient descent raises the positive's similarity and lowers each negative's, in proportion to the softmax weight the loss currently gives it, divided by τ . The rest of this page carries this vector through the similarities to the embeddings.

Problem 2

Compute $\nabla_u c(u, v)$ and $\nabla_v c(u, v)$. Show that $\nabla_u c$ is orthogonal to u and find its norm.

$$c = \hat{u}^\top \hat{v}, \text{ and } \hat{v} \text{ does not depend on } u.$$

Definition of the cosine; v is held fixed.

$$\nabla_u c = \left(\frac{\partial \hat{u}}{\partial u} \right)^\top \hat{v} = \frac{1}{\|u\|} (I - \hat{u} \hat{u}^\top) \hat{v}.$$

$c = a^\top \hat{u}$ with $a = \hat{v}$ constant, so $\nabla_{\hat{u}} c = \hat{v}$ and the chain rule through $\hat{u}(u)$ multiplies by the transposed Jacobian, which is symmetric (Before you start).

$(I - \hat{u}\hat{u}^\top)\hat{v} = \hat{v} - \hat{u}(\hat{u}^\top\hat{v}) = \hat{v} - c\hat{u}.$	Distribute; $\hat{u}^\top\hat{v}$ is the scalar c .
By symmetry, $\nabla_v c = \frac{1}{\ v\ }(\hat{u} - c\hat{v}).$	$c(u, v) = c(v, u)$, so swap the roles.
$u^\top \nabla_u c = \frac{1}{\ u\ } (u^\top \hat{v} - c u^\top \hat{u}) =$ $\frac{1}{\ u\ } (\ u\ c - c\ u\) = 0.$	$u^\top \hat{v} = \ u\ \hat{u}^\top \hat{v} = \ u\ c$ and $u^\top \hat{u} = \ u\ .$
$\ \nabla_u c\ ^2 = \frac{1}{\ u\ ^2} (\hat{v}^\top \hat{v} - 2c\hat{u}^\top \hat{v} +$ $c^2 \hat{u}^\top \hat{u}) = \frac{1 - 2c^2 + c^2}{\ u\ ^2} = \frac{1 - c^2}{\ u\ ^2}.$	Expand $\ \hat{v} - c\hat{u}\ ^2$ with unit vectors and $\hat{u}^\top \hat{v} = c.$
$\nabla_u c = \frac{\hat{v} - c\hat{u}}{\ u\ }, \nabla_v c = \frac{\hat{u} - c\hat{v}}{\ v\ }; u^\top \nabla_u c = 0$ and $\ \nabla_u c\ = \frac{\sqrt{1 - c^2}}{\ u\ }$	The gradient is the part of $\hat{v}$ orthogonal to $\hat{u}$ (the direction that turns u towards v), scaled by $1/\ u\ $: a long anchor turns more slowly. It vanishes when $c = \pm 1$, where u and v are already aligned or opposed, and nothing in it can change the length of u .

Problem 3

Compute $\nabla_u \ell$ for InfoNCE with cosine similarities, and simplify it to an expression in $\hat{u}$, the unit candidates $\hat{v}_j$, the weights p_j and the similarities c_j .

$\nabla_u \ell = \sum_j \frac{\partial \ell}{\partial c_j} \nabla_u c_j = \frac{1}{\tau} \sum_j (p_j - y_j) \nabla_u c_j.$	ℓ depends on u only through the K similarities; chain rule over all of them with Problem 1.
$= \frac{1}{\tau \ u\ } \sum_j (p_j - y_j) (\hat{v}_j - c_j \hat{u}).$	Problem 2 for each c_j ; the common factor $1/\ u\ $ comes out.
$\sum_j (p_j - y_j) \hat{v}_j = \sum_j p_j \hat{v}_j - \hat{v}_1.$	y is one-hot on the positive.
$\sum_j (p_j - y_j) c_j \hat{u} = \left(\sum_j p_j c_j - c_1 \right) \hat{u}.$	The same split; $\hat{u}$ does not depend on j .

$$\nabla_u \ell = \frac{1}{\tau \|u\|} \left[\sum_j p_j \hat{v}_j - \hat{v}_1 - \left(\sum_j p_j c_j - c_1 \right) \hat{u} \right]$$

The step $-\eta \nabla_u \ell$ pulls u towards the positive direction $\hat{v}_1$ and pushes it away from the softmax-weighted average of all candidates, $\sum_j p_j \hat{v}_j$, which is dominated by the negatives it currently confuses with the positive; the $\hat{u}$ term removes whatever component of that lies along u , so that $u^\top \nabla_u \ell = 0$ (Problem 7). The $\sum_j (p_j - y_j) = 0$ identity does not kill the $\hat{u}$ term, because each term carries its own c_j .

Problem 4

Compute $\nabla_{v_j} \ell$ for the positive ($j = 1$) and for a negative ($j \neq 1$), and say in which direction each candidate moves under gradient descent.

ℓ depends on v_j only through c_j .

v_j appears in no other similarity.

$$\nabla_{v_j} \ell = \frac{\partial \ell}{\partial c_j} \nabla_{v_j} c_j = \frac{p_j - y_j}{\tau} \cdot \frac{\hat{u} - c_j \hat{v}_j}{\|v_j\|}.$$

Problem 1 for the first factor and Problem 2 (the ∇_v form) for the second.

Positive: $\nabla_{v_1} \ell = \frac{p_1 - 1}{\tau \|v_1\|} (\hat{u} - c_1 \hat{v}_1)$, with $p_1 - 1 < 0$.

$y_1 = 1$ and $0 < p_1 < 1$.

Negative: $\nabla_{v_j} \ell = \frac{p_j}{\tau \|v_j\|} (\hat{u} - c_j \hat{v}_j)$, with $p_j > 0$.

$y_j = 0$.

$\nabla_{v_j} \ell = \frac{p_j - y_j}{\tau \|v_j\|} (\hat{u} - c_j \hat{v}_j)$: **the positive moves along $+(\hat{u} - c_1 \hat{v}_1)$, towards the anchor's direction; each negative moves along $-(\hat{u} - c_j \hat{v}_j)$, away from it, with weight p_j**

$\hat{u} - c_j \hat{v}_j$ is the component of $\hat{u}$ orthogonal to $\hat{v}_j$, the direction that turns v_j towards u (Problem 2). A negative with tiny p_j , one the model already separates, is barely touched; a negative with p_j close to p_1 is pushed as hard as the positive is pulled. In a batch where every example is also a negative for the others, each v_j collects one such term from every anchor (Problem 9).

Problem 5

Show that the weight the loss puts on negative j relative to negative k is $p_j/p_k = e^{(c_j - c_k)/\tau}$. Describe p and $\nabla_c \ell$ as $\tau \rightarrow 0$ and as $\tau \rightarrow \infty$.

$$\frac{p_j}{p_k} = \frac{e^{c_j/\tau}}{e^{c_k/\tau}} = e^{(c_j - c_k)/\tau}.$$

The common denominator of the softmax cancels.

With $\tau = 0.1$, a negative whose similarity is 0.1 higher than another's gets $e^1 \approx 2.7$ times the weight; with $\tau = 0.01$, $e^{10} \approx 22,000$ times.

Step 1 with $c_j - c_k = 0.1$.

As $\tau \rightarrow 0$, $p \rightarrow e_{j^*}$ where j^* is the candidate with the largest similarity.

The softmax page, Problem 9: $\text{softmax}(c/\tau)$ concentrates on the largest entry as the temperature goes to 0 (assuming a unique maximum).

Then $\nabla_c \ell = (p - y)/\tau \rightarrow (e_{j^*} - e_1)/\tau$: zero if the positive is the top candidate, and otherwise a push on the single hardest negative and a pull on the positive, each of size $1/\tau$.

Problem 1 with step 3. The loss itself tends to $\max(0, \max_{j \neq 1} c_j - c_1)/\tau$, a hinge on the hardest negative.

As $\tau \rightarrow \infty$, $p \rightarrow \frac{1}{K} \mathbf{1}$ and $\nabla_c \ell \rightarrow (\frac{1}{K} \mathbf{1} - y)/\tau \rightarrow 0$.

All logits go to 0, so the softmax is uniform; the $1/\tau$ then sends the gradient to zero.

$p_j/p_k = e^{(c_j - c_k)/\tau}$; $\tau \rightarrow 0$ gives a hinge on the hardest negative with gradient $(e_{j^*} - e_1)/\tau$; $\tau \rightarrow \infty$ gives uniform weights and a vanishing gradient

The temperature is a hardness dial: small τ makes the loss attend to the few negatives nearest the anchor, large τ treats all negatives alike. Typical values (0.05 to 0.1 for SimCLR, about 0.01 for a trained CLIP) sit near the hard end, which is why these losses need many negatives: with few, the hardest negative is often not hard.

Problem 6

The symmetric batch loss used by CLIP is $L = \frac{1}{2N} \sum_{i=1}^N [\ell_i^{\text{row}} + \ell_i^{\text{col}}]$ with $\ell_i^{\text{row}} = -Z_{ii} + \log \sum_j e^{Z_{ij}}$ (anchor u_i against all v_j) and $\ell_i^{\text{col}} = -Z_{ii} + \log \sum_j e^{Z_{ji}}$ (anchor v_i against all u_j). Compute $\nabla_Z L$.

$\partial \ell_i^{\text{row}} / \partial Z_{ij} = (P_r)_{ij} - \delta_{ij}$, and ℓ_i^{row} does not depend on any other row of Z .

Problem 1 applied to row i of Z with label i : its softmax is row i of P_r .

$$\nabla_Z \sum_i \ell_i^{\text{row}} = P_r - I.$$

Stack step 1 over the rows: entry (i, j) gets its contribution from ℓ_i^{row} alone.

$\partial \ell_i^{\text{col}} / \partial Z_{ji} = (P_c)_{ji} - \delta_{ji}$, and ℓ_i^{col} depends only on column i .

The same with column i of Z and label i ; its softmax is column i of P_c .

$$\nabla_Z \sum_i \ell_i^{\text{col}} = P_c - I.$$

Stack over the columns: entry (j, i) comes from ℓ_i^{col} .

$$\nabla_Z L = \frac{1}{2N} (P_r + P_c - 2I)$$

$N \times N$, the shape of Z . Each row of $P_r - I$ and each column of $P_c - I$ sums to 0, so all the entries of $\nabla_Z L$ sum to 0. The diagonal entries $(P_r)_{ii} + (P_c)_{ii} - 2$ are negative (pull the pairs together) and the off-diagonal entries are positive (push every non-pair apart). $P_c \neq P_r^\top$ in general: the two directions normalise over different sets, so neither half can be dropped and recovered "by symmetry".

Problem 7

Show that ℓ is unchanged by $u \rightarrow \alpha u$ for any $\alpha > 0$, that $u^\top \nabla_u \ell = 0$, and that $\nabla_u \ell$ evaluated at αu is $\frac{1}{\alpha}$ times its value at u . If $u = Wx$ for a projection W and input x , compute $\nabla_W \ell$.

$$c(\alpha u, v) = \frac{\alpha u^\top v}{\alpha \|u\| \|v\|} = c(u, v).$$

$\|\alpha u\| = \alpha \|u\|$ for $\alpha > 0$; the α cancels. Every c_j , hence z , p and ℓ , is unchanged.

$\frac{d}{d\alpha} \ell(\alpha u) = u^\top \nabla_u \ell(\alpha u)$, and the left side is 0 for all α .

Chain rule along the ray $\alpha \mapsto \alpha u$; step 1 says ℓ is constant on it. At $\alpha = 1$, $u^\top \nabla_u \ell = 0$.

$\nabla_u \ell = \frac{1}{\tau} \sum_j (p_j - y_j) \nabla_u c_j$, and each $\nabla_u c_j$ at αu is $\frac{\hat{v}_j - c_j \hat{u}}{\alpha \ u\ }$, with $p_j, c_j, \hat{u}$ unchanged.	Problem 3, step 1, and Problem 2 with $\ \alpha u\ = \alpha \ u\ $ and $\widehat{\alpha u} = \hat{u}$.
So $\nabla_u \ell(\alpha u) = \frac{1}{\alpha} \nabla_u \ell(u)$.	Every term is divided by α .
$\nabla_W \ell = (\nabla_u \ell) x^\top$.	$u = Wx$ is linear in W with x fixed: $\partial \ell / \partial W_{ab} = \sum_i (\partial \ell / \partial u_i) (\partial u_i / \partial W_{ab}) = (\nabla_u \ell)_a x_b$, the one-hidden-layer page's weight-gradient pattern.
$\ell(\alpha u) = \ell(u)$; $u^\top \nabla_u \ell = 0$; $\nabla_u \ell(\alpha u) = \frac{1}{\alpha} \nabla_u \ell(u)$; $\nabla_W \ell = (\nabla_u \ell) x^\top$	The loss sees only the direction of u , so its gradient can only turn u , and it turns a long u less: the effective learning rate on the direction falls as $1/\ u\ ^2$ (one factor from the gradient, one from the angle a fixed step subtends). Nothing in the loss shrinks $\ u\ $, and $\ Wx\ $ tends to grow under noisy gradient steps, so a scale-invariant loss relies on weight decay to keep its effective learning rate from decaying (the weight-decay page).

Problem 8

Compute $\partial \ell / \partial \tau$. CLIP learns the temperature through a logit scale $s = 1/\tau = e^t$ with t the trained parameter. Compute $\partial \ell / \partial t$, and say which way t moves when the positive's similarity exceeds the softmax-weighted average similarity.

$\ell = -\frac{c_1}{\tau} + \log \sum_j e^{c_j/\tau}$ with the c_j fixed.	Problem 1's loss with $z_j = c_j/\tau$ written out.
$\frac{\partial}{\partial \tau} \left(-\frac{c_1}{\tau} \right) = \frac{c_1}{\tau^2}$.	$\frac{d}{d\tau} \tau^{-1} = -\tau^{-2}$.
$\frac{\partial}{\partial \tau} \log \sum_j e^{c_j/\tau} = \sum_j p_j \cdot \frac{\partial}{\partial \tau} \frac{c_j}{\tau} = -\frac{1}{\tau^2} \sum_j p_j c_j$.	Chain rule through each logit: $\partial \ell / \partial z_j = p_j$ for the log-sum-exp part (Problem 1, step 2), and $\partial z_j / \partial \tau = -c_j/\tau^2$.

$\frac{\partial \ell}{\partial \tau} = \frac{c_1 - \sum_j p_j c_j}{\tau^2}.$	Add steps 2 and 3.
$\tau = e^{-t}$, so $\frac{d\tau}{dt} = -e^{-t} = -\tau$ and $\frac{\partial \ell}{\partial t} = \frac{\partial \ell}{\partial \tau} \cdot (-\tau) = -\frac{c_1 - \sum_j p_j c_j}{\tau}.$	$s = e^t = 1/\tau$ inverts to $\tau = e^{-t}$; chain rule.
$\frac{\partial \ell}{\partial \tau} = \frac{c_1 - \sum_j p_j c_j}{\tau^2}$ and $\frac{\partial \ell}{\partial t} = \frac{\sum_j p_j c_j - c_1}{\tau};$ when $c_1 > \sum_j p_j c_j,$ $\partial \ell / \partial t < 0$, so gradient descent increases t and decreases τ	When the positive beats the weighted average, sharpening the softmax lowers the loss, so the model learns a smaller temperature; when the positive is losing, it learns a larger one to soften the penalty. Left unbounded, a model that is winning sharpens without limit, which is why CLIP clips the logit scale at 100 ($\tau \geq 0.01$). The $\sum_j p_j c_j$ is the loss's own expectation of the similarity under p .

Problem 9

For the symmetric batch loss of Problem 6 with $Z = \hat{U}\hat{V}^\top/\tau$, compute $\nabla_{\hat{U}}L$ and $\nabla_{\hat{V}}L$ in matrix form, then $\nabla_{u_i}L$ through the normalisation $\hat{u}_i = u_i/\|u_i\|$.

Write $\Gamma = \nabla_Z L = \frac{1}{2N}(P_r + P_c - 2I)$ from Problem 6.

$Z = \frac{1}{\tau}\hat{U}\hat{V}^\top$ has $\hat{U}$ as its left factor and $\hat{V}^\top$ as its right factor.	A matrix product scaled by a constant.
$\nabla_{\hat{U}}L = \frac{1}{\tau}\Gamma\hat{V}, N \times d.$	The attention page's left-factor pattern: for $C = PR$, $\nabla_P L = (\nabla_C L)R^\top$, with $R = \hat{V}^\top$ and the constant $1/\tau$ carried along.
$\nabla_{\hat{V}}L = \frac{1}{\tau}\Gamma^\top\hat{U}, N \times d.$	The right-factor pattern $\nabla_R L = P^\top \nabla_C L$ gives $\nabla_{\hat{V}^\top} L = \frac{1}{\tau}\hat{U}^\top \Gamma$; transpose to get the gradient with respect to $\hat{V}$.

Row i of $\hat{U}$ depends on u_i alone, so

$$\nabla_{u_i} L = \left(\frac{\partial \hat{u}_i}{\partial u_i} \right)^\top (\nabla_{\hat{U}} L)_{i,:}^\top = \frac{1}{\|u_i\|} (I - \hat{u}_i \hat{u}_i^\top) (\nabla_{\hat{U}} L)_{i,:}^\top.$$

Chain rule through the normalisation with the Jacobian from Before you start, which is symmetric; the upstream gradient at $\hat{u}_i$ is row i of step 2, as a column.

$\nabla_{\hat{U}} L = \frac{1}{\tau} \Gamma \hat{V}$, $\nabla_{\hat{V}} L = \frac{1}{\tau} \Gamma^\top \hat{U}$, $\Gamma = \frac{1}{2N} (P_r + P_c - 2I)$; **then** $\nabla_{u_i} L = \frac{1}{\|u_i\|} (I - \hat{u}_i \hat{u}_i^\top) \left(\frac{1}{\tau} \Gamma \hat{V} \right)_{i,:}^\top$, **and likewise for v_i with $\Gamma^\top \hat{U}$**

Row i of $\Gamma \hat{V}$ is $\sum_j \Gamma_{ij} \hat{v}_j$: the positive $\hat{v}_i$ with negative weight, every other $\hat{v}_j$ with positive weight, which is Problem 3 for anchor u_i plus the terms from u_i serving as a negative for the other anchors through P_c . The projection then drops the part along $\hat{u}_i$ and the $1/\|u_i\|$ scales it, exactly as in Problem 2. The whole backward pass is two matrix products and a row-wise projection; the $N \times N$ matrix Γ is the only thing that scales with the batch.

Problem 10

Suppose the logits were the plain dot products $z_j = u^\top v_j / \tau$ with no normalisation. Compute $\nabla_u \ell$ and $u^\top \nabla_u \ell$, and show that when the positive has the largest logit the loss can be reduced by scaling u up, without changing its direction.

$$\nabla_u \ell = \sum_j (p_j - y_j) \nabla_u z_j = \frac{1}{\tau} \sum_j (p_j - y_j) v_j.$$

Problem 1's $\nabla_z \ell = p - y$ and $\nabla_u (u^\top v_j / \tau) = v_j / \tau$.

$$u^\top \nabla_u \ell = \frac{1}{\tau} \sum_j (p_j - y_j) u^\top v_j = \sum_j (p_j - y_j) z_j = \sum_j p_j z_j - z_1.$$

$u^\top v_j / \tau = z_j$, and y picks out z_1 .

$$\frac{d}{d\alpha} \ell(\alpha u) \Big|_{\alpha=1} = u^\top \nabla_u \ell = \sum_j p_j z_j - z_1.$$

Chain rule along the ray, as in Problem 7, step 2. Here it is not zero.

If $z_1 > z_j$ for every $j \neq 1$ then $\sum_j p_j z_j < z_1$, so the derivative is negative.

A weighted average of the z_j with weights $p_j > 0$ summing to 1 is strictly less than the largest of them when the others are smaller.

The same holds at every $\alpha > 0$, since $z_j(\alpha u) = \alpha z_j(u)$ keeps the ordering.	Scaling u scales every logit by the same positive factor; the positive stays on top and step 4 applies again.
$\nabla_u \ell = \frac{1}{\tau} \sum_j (p_j - y_j) v_j, u^\top \nabla_u \ell = \sum_j p_j z_j - z_1$; when the positive leads, $\ell(\alpha u)$ decreases in α, so the loss rewards growing $\ u\$	With dot-product logits, the encoder can drive $\ell \rightarrow 0$ on every example it already gets right by inflating embedding norms, which sharpens the softmax without learning anything new about directions; the gradient has a component along u that does exactly this. Normalising removes that component (Problem 7) and the temperature puts the sharpness back under control (Problem 5): cosine plus τ is dot product with the norm fixed at $1/\tau$.

Where this goes wrong

1. Cosine gradient as a dot-product gradient

Once $\hat{u}$ and $\hat{v}$ are computed, $c = \hat{u}^\top \hat{v}$ looks like a dot product with two constant vectors.

$c(u, v) = \hat{u}^\top \hat{v}$	Right so far.
“$\nabla_u (u^\top a) = a$, and here $a = \hat{v}/\ u\$.”	The shortcut that causes the mistake: $\ u\$ in the denominator is treated as a constant, when it is a function of u with gradient $\hat{u}$.
$\nabla_u c = \frac{\hat{v}}{\ u\ } \times$	The missing term is $-\frac{c \hat{u}}{\ u\ }$ (Problem 2), the part of the answer that makes $u^\top \nabla_u c = 0$. The wrong gradient has $u^\top \nabla_u c = c$, so a descent step changes $\ u\$: it shrinks the anchor when the positive is similar and grows it when a negative is, and the scale invariance of Problem 7 is gone. The two agree only when $c = 0$.

2. Dropping the $1/\tau$

The temperature is a constant, and $p - y$ is the gradient everyone remembers.

$\nabla_z \ell = p - y$	Right so far: Problem 1.
“The similarities are the logits up to a constant, so their gradient is the same.”	The shortcut that causes the mistake: a constant factor in the forward pass is a constant factor in the backward pass.
$\nabla_c \ell = p - y$ ✗	It is $(p - y)/\tau$ (Problem 1), and with $\tau = 0.07$ the gradient reaching the encoder is 14 times too small. A fixed τ makes this a learning-rate error on the encoder only, while a learned temperature (Problem 8) and any other loss terms keep their true scale, so the balance between them is wrong and no learning-rate sweep fixes it.

3. Symmetric loss computed in one direction

Half the CLIP loss is a cross-entropy over rows of Z , and the other half looks like the same thing again.

$\nabla_Z \sum_i \ell_i^{\text{row}} = P_r - I$	Right so far: Problem 6, step 2.
“The column direction is the same loss with the roles swapped, so it gives the same gradient; just double the row loss.”	The assumption that causes the mistake: the row softmax normalises each u_i over all v_j , the column softmax each v_i over all u_j ; the two sets of weights are different.
$\nabla_Z L = \frac{1}{N} (P_r - I)$ ✗	The gradient is $\frac{1}{2N} (P_r + P_c - 2I)$ (Problem 6), and $P_c \neq P_r^\top$. With rows only, a text v_j is never an anchor: it is pushed by the images' softmaxes but never gets its own competition among images, so a text that is similar to many images is never penalised for it. Training runs, and the retrieval in the text-to-image direction is measurably worse.

4. Normalisation backward without the projection

$\hat{u} = u/\|u\|$ is a division by a number, and dividing the upstream gradient by the same number looks like the backward pass.

$\nabla_{\hat{u}} \ell = \frac{1}{\tau} \sum_j (p_j - y_j) \hat{v}_j$	Right so far: the gradient at the unit vector, before the normalisation.
“$\hat{u} = u/\ u\$, so $\nabla_u \ell = \nabla_{\hat{u}} \ell / \ u\$.”	The shortcut that causes the mistake: $\ u\$ is held fixed, so the Jacobian $\frac{1}{\ u\ } (I - \hat{u} \hat{u}^\top)$ loses its projection.
$\nabla_u \ell = \frac{1}{\ u\ } \cdot \frac{1}{\tau} \sum_j (p_j - y_j) \hat{v}_j \quad \times$	The true gradient is this with $(I - \hat{u} \hat{u}^\top)$ applied (Problem 9, step 4), which subtracts the component along $\hat{u}$: $\frac{1}{\tau \ u\ } (\sum_j p_j c_j - c_1) \hat{u}$ (Problem 3). The wrong gradient is not orthogonal to u; when the positive leads it points along $-u$, so the step grows the anchor, and the norm drift Problem 10 warned about comes back through the backward pass even though the forward pass normalises.

5. Temperature gradient with the wrong sign

The learned parameter is the logit scale $s = e^t$, and the derivative with respect to τ is the one that is easy to write down.

$\frac{\partial \ell}{\partial \tau} = \frac{c_1 - \sum_j p_j c_j}{\tau^2}$	Right so far: Problem 8, step 4.
“t parametrises the temperature, so update t with the temperature's gradient.”	The slip that causes the mistake: $\tau = e^{-t}$ decreases in t, and the chain rule carries the factor $d\tau/dt = -\tau$.

$$\frac{\partial \ell}{\partial t} = \frac{c_1 - \sum_j p_j c_j}{\tau^2} \times$$

The gradient is $-\tau$ times this, $\frac{\sum_j p_j c_j - c_1}{\tau}$ (Problem 8): opposite sign and a different scale. With the wrong sign, a model whose positives are winning softens its softmax instead of sharpening it, and t runs away in the wrong direction until the clip at $s = 100$ or at $s \rightarrow 0$ stops it. Autodiff does not make this mistake; a hand-written update, or a schedule that adjusts τ directly from $\partial \ell / \partial \tau$ while the model stores t , does.

eulerpad.app/practice/contrastive-loss-and-infonce-gradient