# Attention backward

Ten problems on the backward pass of scaled dot-product attention: the gradients to V, A, the scores, Q and K, the projections in self-attention, the causal mask, why the scores are divided by √d, and the multi-head output projection, with worked solutions and the mistakes that transpose the wrong matrix.

**BEFORE YOU START**

Scaled dot-product attention is three matrix products and a softmax: scores from queries and keys, weights from the scores, outputs from the weights and the values. Its backward pass is the same pieces run in reverse, and every step is either a matrix product, whose gradient is another matrix product with one factor transposed, or the softmax page's row result applied to each row. These ten problems derive each gradient, push them through the projections of self-attention, add the causal mask and the $1/\sqrt{d}$ scaling, and finish with the multi-head output projection. The five mistakes are the ones that give a plausible shape or a plausible number: a missing transpose, a softmax Jacobian that mixes rows, a dropped $1/\sqrt{d}$, a path through $X$ left out, and a mask applied after the softmax.

- The conventions are those of the previous pages: vectors are columns, Jacobians are in numerator layout, a gradient has the shape of the variable it is taken with respect to, and $\odot$ is the elementwise product.
- One head. $Q \in \mathbb{R}^{n\times d}$, $K \in \mathbb{R}^{m\times d}$ and $V \in \mathbb{R}^{m\times d_v}$, with rows as positions: $n$ query positions, $m$ key positions, width $d$ for queries and keys and $d_v$ for values.
- $S = QK^\top/\sqrt{d}$ is $n \times m$; $A = \mathrm{softmax}(S)$ is taken row by row; $O = AV$ is $n \times d_v$.
- $L$ is a scalar loss that depends on $Q$, $K$ and $V$ only through $O$. The upstream gradient is $G = \nabla_O L$, and $\tilde{G} := \nabla_A L$.
- Indices: $i$ is a query position (a row of $Q$, $S$, $A$, $O$), $j$ a key position (a row of $K$ and $V$, a column of $S$ and $A$), $l$ a column of $Q$ and $K$, and $c$ a column of $V$ and $O$.
- $\mathbf{1}$ is the all-ones vector of whatever length the product needs. $\mathrm{rowsum}(B) := B\mathbf{1}$ is a column holding the row sums of a matrix $B$, so $\mathrm{rowsum}(B)\,\mathbf{1}^\top$ copies each row's sum across that row.
- The softmax page's row result: for one row, $a = \mathrm{softmax}(s)$ with upstream $\tilde{g} = \nabla_a L$, the Jacobian $J = \mathrm{diag}(a) - aa^\top$ is symmetric (the softmax page, Problem 3), so $\nabla_s L = J\tilde{g} = a \odot \tilde{g} - a\,(a^\top\tilde{g})$, that is, $\nabla_s = a \odot (\tilde{g} - (a^\top\tilde{g})\mathbf{1})$.
- Every matrix below has rows as positions, so every formula is the row formula stacked.
- Self-attention: $Q = XW_Q$, $K = XW_K$, $V = XW_V$ with $X \in \mathbb{R}^{n\times d_{\text{model}}}$, so $m = n$. $H$ is the number of heads and $h$ indexes them.

Builds on: Jacobians and the chain rule, The softmax Jacobian

## Problems

**1.** Give the shapes of $S$, $A$ and $O$, and the number of multiply-adds to form $S$. What does that cost look like as the sequence length $n = m$ grows? difficulty ·

**2.** Compute $\nabla_V L$. difficulty ··

**3.** Compute $\tilde{G} = \nabla_A L$. difficulty ··

**4.** Compute $\nabla_S L$ from $\tilde{G}$ and $A$, using the softmax Jacobian row by row, and write it as one matrix expression with no $m \times m$ Jacobians. difficulty ···

**5.** Compute $\nabla_Q L$. difficulty ··

**6.** Compute $\nabla_K L$. difficulty ··

**7.** Self-attention: $Q = XW_Q$, $K = XW_K$, $V = XW_V$. Compute $\nabla_{W_Q} L$, $\nabla_{W_K} L$, $\nabla_{W_V} L$ and $\nabla_X L$. difficulty ···

**8.** A causal mask sets $S_{ij} = -\infty$ for $j > i$ before the softmax. What are the masked entries of $A$, and what are the masked entries of $\nabla_S L$? Show it from Problem 4. difficulty ··

**9.** If the entries of $q$ and $k$ are independent with mean 0 and variance 1, what is $\mathrm{Var}(q^\top k)$? What does dividing by $\sqrt{d}$ achieve, and where does the factor appear in the backward pass? difficulty ··

**10.** Multi-head: $O = [O_1 \ \cdots \ O_H]\, W_O$ with $O_h \in \mathbb{R}^{n \times d_v}$ and $W_O \in \mathbb{R}^{Hd_v \times d_{\text{model}}}$. Compute $\nabla_{W_O} L$ and $\nabla_{O_h} L$. difficulty ···

## Answers

1. $S, A : n \times m$; $O : n \times d_v$; $S$ costs $nmd$ multiply-adds, quadratic in the sequence length
2. $\nabla_V L = A^\top G$ $(m \times d_v)$
3. $\tilde{G} = \nabla_A L = G V^\top$ $(n \times m)$
4. $\nabla_S L = A \odot \left(\tilde{G} - \operatorname{rowsum}(A \odot \tilde{G})\, \mathbf{1}^\top\right)$
5. $\nabla_Q L = \frac{1}{\sqrt{d}} (\nabla_S L) K$ $(n \times d)$
6. $\nabla_K L = \frac{1}{\sqrt{d}} (\nabla_S L)^\top Q$ $(m \times d)$
7. $\nabla_{W_Q} L = X^\top \nabla_Q L$; $\nabla_{W_K} L = X^\top \nabla_K L$; $\nabla_{W_V} L = X^\top \nabla_V L$; $\nabla_X L = \nabla_Q L\, W_Q^\top + \nabla_K L\, W_K^\top + \nabla_V L\, W_V^\top$
8. masked $A_{ij} = 0$, and since $\nabla_S L = A \odot (\cdots)$, masked $(\nabla_S L)_{ij} = 0$ exactly; the unmasked entries follow Problem 4 with the masked columns contributing nothing to the row sums
9. $\operatorname{Var}(q^\top k) = d$; dividing by $\sqrt{d}$ makes the scores order 1 so the softmax does not saturate; the same $\frac{1}{\sqrt{d}}$ multiplies $\nabla_Q L$ and $\nabla_K L$ (Problems 5–6) and nothing else
10. $\nabla_{W_O} L = [O_1 \ \cdots \ O_H]^\top G$; $\nabla_{O_h} L$ = columns $(h-1)d_v + 1$ to $h d_v$ of $G W_O^\top$

## Worked solutions

### Problem 1

> Give the shapes of $S$, $A$ and $O$, and the number of multiply-adds to form $S$. What does that cost look like as the sequence length $n = m$ grows?

| | |
|---|---|
| $QK^\top$ is $(n \times d)(d \times m) = n \times m$, and so is $S$. | $K^\top$ is $d \times m$, and dividing by the scalar $\sqrt{d}$ keeps the shape. |
| $A$ is $n \times m$. | The softmax is applied to each row separately and keeps each row's length. |
| $O = AV$ is $(n \times m)(m \times d_v) = n \times d_v$. | The inner dimensions agree: each query position gets a weighted sum of the $m$ value rows. |
| $S_{ij} = \frac{1}{\sqrt{d}} \sum_l Q_{il} K_{jl}$ takes $d$ multiply-adds, and there are $nm$ entries. | Each score is the dot product of query row $i$ with key row $j$, both of length $d$. |
| $S, A : n \times m$; $O : n \times d_v$; $S$ **costs** $nmd$ **multiply-adds, quadratic in the sequence length** | With $n = m$ the cost is $n^2 d$: doubling the sequence length quadruples it. $O = AV$ costs another $nmd_v$, and $A$ itself has $n^2$ entries to store for the backward pass, so time and memory both grow quadratically. |

### Problem 2

> Compute $\nabla_V L$.

| | |
|---|---|
| $O_{ic} = \sum_j A_{ij} V_{jc}$. | Index form of $O = AV$, which is linear in $V$ for fixed $A$. |
| $V_{jc}$ appears in $O_{ic}$ for every $i$, with coefficient $A_{ij}$, and in no entry of another column. | Column $c$ of $O$ is built from column $c$ of $V$ only, and every query row uses key row $j$. |

| | |
|---|---|
| $\partial L/\partial V_{jc} = \sum_i G_{ic} A_{ij}$. | $L$ depends on $V$ only through $O$; by step 2 the chain rule sums over the rows $i$ of $O$ in column $c$. |
| $\sum_i A_{ij} G_{ic} = (A^\top G)_{jc}$, with shapes $(m \times n)(n \times d_v) = m \times d_v$. | The sum runs over $A$'s row index, which a product can only contract if $A$ is transposed. |
| $\nabla_V L = A^\top G$ **(**$m \times d_v$**)** | The shape of $V$. The pattern is general: for $C = PR$, the gradient with respect to the right factor is $P^\top \nabla_C L$. |

## Problem 3

> Compute $\tilde{G} = \nabla_A L$.

| | |
|---|---|
| $A_{ij}$ appears in $O_{ic}$ for every $c$, with coefficient $V_{jc}$, and in no other row of $O$. | Problem 2, step 1: row $i$ of $A$ builds only row $i$ of $O$. |
| $\partial L/\partial A_{ij} = \sum_c G_{ic} V_{jc}$. | $L$ depends on $A$ only through $O$, and step 1 limits the sum to row $i$. |
| $\sum_c G_{ic} V_{jc} = (GV^\top)_{ij}$, with shapes $(n \times d_v)(d_v \times m) = n \times m$. | The sum runs over $V$'s column index, so $V$ is transposed. |
| $\tilde{G} = \nabla_A L = GV^\top$ **(**$n \times m$**)** | The shape of $A$. The companion pattern to Problem 2: for $C = PR$, the gradient with respect to the left factor is $(\nabla_C L) R^\top$. |

## Problem 4

> Compute $\nabla_S L$ from $\tilde{G}$ and $A$, using the softmax Jacobian row by row, and write it as one matrix expression with no $m \times m$ Jacobians.

Write $s_i$, $a_i$ and $\tilde{g}_i \in \mathbb{R}^m$ for row $i$ of $S$, $A$ and $\tilde{G}$, as columns.

| | |
|---|---|
| $a_i = \mathrm{softmax}(s_i)$, and $a_i$ depends on no other row of $S$. | The softmax is taken row by row, so row $i$ of $S$ reaches $L$ only through row $i$ of $A$. |

| | |
|---|---|
| $\nabla_{s_i} L = (\operatorname{diag}(a_i) - a_i a_i^\top)\,\tilde{g}_i = a_i \odot (\tilde{g}_i - (a_i^\top \tilde{g}_i)\mathbf{1})$. | The softmax page's row result with $a = a_i$ and upstream $\nabla_{a_i} L = \tilde{g}_i$, row $i$ of $\tilde{G}$; by step 1, $a_i$ is the only path from $s_i$ to $L$, so no other row's upstream enters. |
| $a_i^\top \tilde{g}_i = \sum_j A_{ij}\tilde{G}_{ij}$ is entry $i$ of $\operatorname{rowsum}(A \odot \tilde{G})$, $n \times 1$. | A dot product of two rows is the sum along that row of their elementwise product. |
| Row $i$ of $\tilde{G} - \operatorname{rowsum}(A \odot \tilde{G})\,\mathbf{1}^\top$ is $\big(\tilde{g}_i - (a_i^\top \tilde{g}_i)\mathbf{1}\big)^\top$, with shapes $(n \times 1)(1 \times m) = n \times m$. | The outer product with $\mathbf{1}^\top$ copies entry $i$ of the column across row $i$, so each row subtracts its own scalar. |
| $\nabla_S L = A \odot \big(\tilde{G} - \operatorname{rowsum}(A \odot \tilde{G})\,\mathbf{1}^\top\big)$ | $n \times m$, the shape of $S$: step 2 stacked, since an elementwise product acts row by row. Each row's Jacobian would be $m \times m$, and the whole matrix's $nm \times nm$; neither is built, and the cost is $O(nm)$. In array code the rowsum is a sum along the last axis with the dimension kept, so it broadcasts. |

## Problem 5

> Compute $\nabla_Q L$.

| | |
|---|---|
| $S_{ij} = \frac{1}{\sqrt{d}} \sum_l Q_{il} K_{jl}$. | Index form of $S = QK^\top/\sqrt{d}$, which is linear in $Q$ for fixed $K$. |
| $Q_{il}$ appears in $S_{ij}$ for every $j$, with coefficient $K_{jl}/\sqrt{d}$, and in no other row of $S$. | Query row $i$ is dotted with every key row, and builds only row $i$ of $S$. |
| $\partial L/\partial Q_{il} = \frac{1}{\sqrt{d}} \sum_j (\nabla_S L)_{ij} K_{jl} = \frac{1}{\sqrt{d}} \big((\nabla_S L) K\big)_{il}$. | $L$ depends on $Q$ only through $S$; step 2 limits the chain rule to row $i$. The sum runs over $K$'s row index, which is also the column index of $\nabla_S L$, so no transpose is needed. |

| | |
|---|---|
| $\nabla_Q L = \frac{1}{\sqrt{d}}(\nabla_S L)K$ $(n \times d)$ | $(n \times m)(m \times d)$, the shape of $Q$. Problem 3's left-factor pattern with $C = S$, $P = Q/\sqrt{d}$ and $R = K^\top$ gives $\nabla_P L = (\nabla_S L)K$; since $P = Q/\sqrt{d}$, each $Q_{il}$ enters $L$ through $P_{il}$ times $\frac{1}{\sqrt{d}}$, so $\nabla_Q L = \frac{1}{\sqrt{d}}\nabla_P L$. |

## Problem 6

> Compute $\nabla_K L$.

| | |
|---|---|
| $K_{jl}$ appears in $S_{ij}$ for every $i$, with coefficient $Q_{il}/\sqrt{d}$, and in no other column of $S$. | Problem 5, step 1: key row $j$ is dotted with every query row and builds only column $j$ of $S$. |
| $\partial L/\partial K_{jl} = \frac{1}{\sqrt{d}} \sum_i (\nabla_S L)_{ij} Q_{il}$. | $L$ depends on $K$ only through $S$, and step 1 limits the chain rule to column $j$. |
| $\sum_i (\nabla_S L)_{ij} Q_{il} = \left((\nabla_S L)^\top Q\right)_{jl}$, with shapes $(m \times n)(n \times d) = m \times d$. | The sum runs over the row index of $\nabla_S L$, so it is transposed. |
| $\nabla_K L = \frac{1}{\sqrt{d}}(\nabla_S L)^\top Q$ $(m \times d)$ | The shape of $K$. Equivalently, $S^\top = KQ^\top/\sqrt{d}$ has $K$ as its left factor, and $\nabla_{S^\top} L = (\nabla_S L)^\top$. |

## Problem 7

> Self-attention: $Q = XW_Q$, $K = XW_K$, $V = XW_V$. Compute $\nabla_{W_Q} L$, $\nabla_{W_K} L$, $\nabla_{W_V} L$ and $\nabla_X L$.

Here $m = n$, $W_Q, W_K \in \mathbb{R}^{d_{\text{model}} \times d}$ and $W_V \in \mathbb{R}^{d_{\text{model}} \times d_v}$. $\nabla_Q L$, $\nabla_K L$ and $\nabla_V L$ are Problems 5, 6 and 2, each taken with the other two inputs held fixed.

| | |
|---|---|
| $\nabla_{W_Q} L = X^\top \nabla_Q L$, $(d_{\text{model}} \times n)(n \times d) = d_{\text{model}} \times d$. | $W_Q$ reaches $L$ only through $Q = XW_Q$, where it is the right factor: Problem 2's pattern. |
| $\nabla_{W_K} L = X^\top \nabla_K L$ and $\nabla_{W_V} L = X^\top \nabla_V L$, $d_{\text{model}} \times d$ and $d_{\text{model}} \times d_v$. | The same argument for $K = XW_K$ and $V = XW_V$; each weight matrix feeds one projection only. |

| | |
|---|---|
| The part of $\nabla_X L$ through $Q$ is $\nabla_Q L\, W_Q^\top$, $(n \times d)(d \times d_{\text{model}}) = n \times d_{\text{model}}$. | In $Q = XW_Q$, $X$ is the left factor: Problem 3's pattern. |
| The parts through $K$ and $V$ are $\nabla_K L\, W_K^\top$ and $\nabla_V L\, W_V^\top$, both $n \times d_{\text{model}}$. | The same pattern; $\nabla_K L$ is $n \times d$ because $m = n$. |
| $\nabla_{W_Q} L = X^\top \nabla_Q L$; $\nabla_{W_K} L = X^\top \nabla_K L$; $\nabla_{W_V} L = X^\top \nabla_V L$; $\nabla_X L = \nabla_Q L\, W_Q^\top + \nabla_K L\, W_K^\top + \nabla_V L\, W_V^\top$ | $X$ feeds all three projections, and the chain rule adds the contributions of every path from a variable to $L$. Each gradient has the shape of its variable. |

## Problem 8

> A causal mask sets $S_{ij} = -\infty$ for $j > i$ before the softmax. What are the masked entries of $A$, and what are the masked entries of $\nabla_S L$? Show it from Problem 4.

Here $m = n$, so $S$ is square and the mask keeps the diagonal and everything below it.

| | |
|---|---|
| $A_{ij} = e^{S_{ij}} \big/ \sum_{j' \le i} e^{S_{ij'}}$ for every $j$. | Row-wise softmax; the masked terms of the denominator are $e^{-\infty} = 0$, so only $j' \le i$ remain, and the $j' = i$ term makes the sum positive. |
| $A_{ij} = 0$ for $j > i$, and the unmasked entries of each row sum to 1. | The numerator is $e^{-\infty} = 0$; the softmax renormalizes over the positions a query may see. |
| $(\nabla_S L)_{ij} = A_{ij}\big(\tilde{G}_{ij} - \sum_{j'} A_{ij'} \tilde{G}_{ij'}\big)$. | Entry $(i, j)$ of Problem 4's formula. |
| For $j > i$ the bracket is finite and the factor $A_{ij}$ is 0, so $(\nabla_S L)_{ij} = 0$. | $\tilde{G} = GV^\top$ is finite, so the bracket is too; zero times a finite number is exactly zero, not a small number. |
| The row sum $\sum_{j'} A_{ij'} \tilde{G}_{ij'}$ has zero terms at the masked $j'$. | Step 2: those $A_{ij'}$ are 0. |

| | |
|---|---|
| **masked $A_{ij} = 0$, and since $\nabla_S L = A \odot (\cdots)$, masked $(\nabla_S L)_{ij} = 0$ exactly; the unmasked entries follow Problem 4 with the masked columns contributing nothing to the row sums** | So no gradient reaches $Q$ or $K$ through a masked score: in Problems 5 and 6 those entries of $\nabla_S L$ multiply by zero. In code the $-\infty$ is often a large negative number such as $-10^9$; its exponential underflows to 0 in floating point, so the same holds. |

## Problem 9

> If the entries of $q$ and $k$ are independent with mean 0 and variance 1, what is $\mathrm{Var}(q^\top k)$? What does dividing by $\sqrt{d}$ achieve, and where does the factor appear in the backward pass?

Here $q, k \in \mathbb{R}^d$ are one query row and one key row as columns, so $q^\top k = \sum_l q_l k_l$ is a score before scaling.

| | |
|---|---|
| $\mathbb{E}[q_l k_l] = \mathbb{E}[q_l]\,\mathbb{E}[k_l] = 0.$ | $q_l$ and $k_l$ are independent, so the expectation of the product factors, and each mean is 0. |
| $\mathrm{Var}(q_l k_l) = \mathbb{E}[q_l^2 k_l^2] - 0^2 = \mathbb{E}[q_l^2]\,\mathbb{E}[k_l^2] = 1.$ | Independence again; $\mathbb{E}[q_l^2] = \mathrm{Var}(q_l) + \mathbb{E}[q_l]^2 = 1$, and likewise for $k_l$. |
| $\mathrm{Var}\left(\sum_l q_l k_l\right) = \sum_l \mathrm{Var}(q_l k_l) = d.$ | The terms for different $l$ are functions of disjoint sets of independent entries, so they are independent and their variances add. |
| $\mathrm{Var}(q^\top k/\sqrt{d}) = d/d = 1.$ | Scaling a random variable by $\lambda$ scales its variance by $\lambda^2$, here $\lambda^2 = 1/d$. |
| Without the scaling the scores have standard deviation $\sqrt{d}$, 8 at $d = 64$, so the softmax of a row is concentrated on a few keys, often one. | The softmax weights of two scores differ by the factor $\exp(\Delta)$, where $\Delta$ is the gap between them. At this spread the top two scores of a row are typically 2 to 3 units apart, and $\Delta = 3$ already gives $\exp(3) \approx 20$. |

| | |
|---|---|
| The more concentrated a row $a$, the smaller $\mathrm{diag}(a) - aa^\top$, so $\nabla_S L$ shrinks and less gradient reaches $Q$ and $K$. | At a one-hot row, a standard basis vector $u$, $\mathrm{diag}(u) - uu^\top = 0$, and the entries $a_j(1 - a_j)$ and $-a_j a_{j'}$ go to 0 as $a$ approaches $u$. |
| In the backward pass $\sqrt{d}$ appears only where $S$ is differentiated with respect to $Q$ and $K$. | $\nabla_V L$, $\tilde{G}$ and $\nabla_S L$ are computed from $A$, $G$ and $V$ (Problems 2 to 4), and none of their formulas contains $d$. |
| $\mathrm{Var}(q^\top k) = d$; **dividing by $\sqrt{d}$ makes the scores order 1 so the softmax does not saturate; the same $\frac{1}{\sqrt{d}}$ multiplies $\nabla_Q L$ and $\nabla_K L$ (Problems 5–6) and nothing else** | The projection gradients of Problem 7 inherit it through $\nabla_Q L$ and $\nabla_K L$ and add no second factor. Trained queries and keys are not independent unit-variance vectors, but the scaling keeps the scores at order 1 at initialization, where saturation would stop learning before it starts. |

### Problem 10

> Multi-head: $O = [O_1 \ \cdots \ O_H]\, W_O$ with $O_h \in \mathbb{R}^{n \times d_v}$ and $W_O \in \mathbb{R}^{Hd_v \times d_{\text{model}}}$. Compute $\nabla_{W_O} L$ and $\nabla_{O_h} L$.

In this problem $O$ is the output of the multi-head layer, $n \times d_{\text{model}}$, $G = \nabla_O L$ has the same shape, and $O_h$, for $h = 1, \ldots, H$, is head $h$'s output from the earlier problems. Write $O_{\text{cat}} = [O_1 \ \cdots \ O_H]$, $n \times Hd_v$.

| | |
|---|---|
| $\nabla_{W_O} L = O_{\text{cat}}^\top G$, $(Hd_v \times n)(n \times d_{\text{model}}) = Hd_v \times d_{\text{model}}$. | $O = O_{\text{cat}} W_O$ with $W_O$ the right factor: Problem 2's pattern. |
| $\nabla_{O_{\text{cat}}} L = G W_O^\top$, $(n \times d_{\text{model}})(d_{\text{model}} \times Hd_v) = n \times Hd_v$. | $O_{\text{cat}}$ is the left factor: Problem 3's pattern. |
| Column $(h-1)d_v + c$ of $O_{\text{cat}}$ is column $c$ of $O_h$, for $c = 1, \ldots, d_v$. | Concatenation places $O_1$ in columns 1 to $d_v$, $O_2$ in columns $d_v + 1$ to $2d_v$, and so on; each entry of $O_h$ is exactly one entry of $O_{\text{cat}}$. |

| | |
|---|---|
| $\nabla_{W_O} L = [O_1 \ \cdots \ O_H]^\top G$; $\nabla_{O_h} L =$ **columns** $(h-1)d_v + 1$ **to** $hd_v$ **of** $GW_O^\top$ | $n \times d_v$, the shape of $O_h$. Concatenation copies entries without mixing them, so its backward pass slices the gradient into the same blocks. From there each head runs Problems 2 to 6 on its own block, and the heads' contributions to $X$ add, as the three paths did in Problem 7. |

## Where this goes wrong

### 1. Gradient to V without the transpose

$O = AV$ puts $A$ on the left, and it is tempting to keep it there on the way back.

| | |
|---|---|
| $O_{ic} = \sum_j A_{ij} V_{jc}$ and $G = \nabla_O L$ | Right so far: Problem 2, step 1. |
| "$V$ is multiplied by $A$, so its gradient is $A$ times the gradient coming back." | The shortcut that causes the mistake: the scalar rule, where the derivative of $av$ is $a$, applied to matrices without asking which index the sum runs over. |
| $\nabla_V L = AG$ ✗ | The shapes $(n \times m)(n \times d_v)$ do not multiply unless $n = m$, and even then it is wrong. The index form, $\partial L / \partial V_{jc} = \sum_i A_{ij} G_{ic}$, puts the sum on $A$'s row index, so the product is $A^\top G$. In self-attention $n = m$, so the wrong version runs and trains on the wrong gradient. |

### 2. Softmax Jacobian applied to the whole matrix

The softmax page's Jacobian is written for one vector, and flattening a matrix into a vector makes it look as though it applies at once. Write $\mathrm{vec}(B)$ for the column of length $nm$ that stacks the rows of an $n \times m$ matrix $B$ one after another (row-major).

| | |
|---|---|
| $\partial a / \partial s = \mathrm{diag}(a) - aa^\top$ for $a = \mathrm{softmax}(s)$ | Right so far: the softmax page, Problem 3, for one row. |
| "Flatten $S$ and $A$ to vectors and use the same Jacobian." | The shortcut that causes the mistake: treating the row-wise softmax as one softmax over all $nm$ scores. |

| | |
|---|---|
| $\nabla_S L = (\mathrm{diag}(a) - aa^\top)\tilde{g}$ with $a = \mathrm{vec}(A)$ and $\tilde{g} = \mathrm{vec}(\tilde{G})$ ✗ | The softmax is per row: entries in different rows do not interact, so the true $nm \times nm$ Jacobian is block-diagonal, with the row Jacobians $\mathrm{diag}(a_i) - a_i a_i^\top$ as blocks, and the row formula of Problem 4 is all there is. The flattened version subtracts $a\,(a^\top \tilde{g})$, where $a^\top \tilde{g}$ sums over every row, so each row is corrected by the whole matrix's total instead of its own. |

### 3. Losing the 1/√d on the way back

The scaling is a fixed number with no parameter in it, and it is easy to decide that the backward pass can ignore it.

| | |
|---|---|
| $S = QK^\top/\sqrt{d}$ and $\nabla_S L$ from Problem 4 | Right so far. |
| "$1/\sqrt{d}$ is a constant, not something we train, so it needs no gradient." | The analogy that causes the mistake: treating the scaling as part of the data, when it is part of the function $S$ of $Q$ and $K$. A constant needs no gradient of its own, but it still multiplies the gradient of everything it scales. |
| $\nabla_Q L = (\nabla_S L)K$ ✗ | $S$ contains the factor, so its derivative with respect to $Q$ does too: $\frac{1}{\sqrt{d}}(\nabla_S L)K$ (Problem 5). The error makes $\nabla_Q L$ (and $\nabla_K L$, if the factor is dropped there too) $\sqrt{d}$ times too large, 8 times at $d = 64$, which under plain SGD acts like a larger learning rate for $W_Q$ (and $W_K$) and passes a check that only compares directions. |

### 4. Only one path from X

Problem 7 has one input and three projections, and it is easy to follow only the one the problem starts with.

| | |
|---|---|
| $\nabla_Q L\, W_Q^\top$ is the gradient reaching $X$ through $Q$ | Right so far: Problem 7, step 3. |
| "$X$ is the query input, so its gradient comes back through the queries." | The analogy that causes the mistake: cross-attention, where the queries come from $X$ and the keys and values from another sequence, so the query path really is the only one. |
| $\nabla_X L = \nabla_Q L\, W_Q^\top$ ✗ | In self-attention $X$ enters through $Q$, $K$ and $V$, and the chain rule sums over every path: the terms $\nabla_K L\, W_K^\top$ and $\nabla_V L\, W_V^\top$ are missing. The shape is right, so nothing fails; the layers below simply train on part of their gradient. |

### 5. Masking A instead of S

Problem 8 shows the masked weights are zero, and zeroing them directly looks like a shortcut to the same place. Write $M$ for the $n \times n$ matrix with $M_{ij} = 1$ for $j \le i$ and 0 otherwise.

| | |
|---|---|
| The causal mask must make $A_{ij} = 0$ for $j > i$ | Right so far: Problem 8, step 2. |
| "Compute the full softmax, then zero the entries a query may not see." | The shortcut that causes the mistake: masking the output of the softmax instead of its input. |
| $A \leftarrow A \odot M$ after the softmax ✗ | Rows no longer sum to 1, so the forward pass is wrong before the backward pass starts. Worse, each denominator still contains $e^{S_{ij}}$ for the future positions $j > i$, so the kept weights depend on later keys and gradient flows into them: the model can see the future. Mask the scores with $-\infty$ and the softmax renormalizes. |

**More at eulerpad.app.** This is one of 82 free practice sets with worked solutions at eulerpad.app/practice, alongside guides to the topics behind them.

**EulerPad**, the app, reads your handwritten math on iPad, iPhone and Android and checks it line by line. It's free to start.

This set is also Chapter 17 of *Deep Learning Math by Hand*, a 250-problem workbook with worked solutions: eulerpad.app/workbooks

eulerpad.app/practice/attention-backward