Batch norm backward

Ten problems on the backward pass of batch normalization: the gradients of γ, β and the batch statistics, the compact input gradient and why each feature's column of it sums to zero, batch norm as layer norm on the transpose, inference with running statistics, a batch of one and the role of ε, with worked solutions and the mistakes that hold the batch statistics constant or take them over the wrong axis.

BEFORE YOU START

Batch normalization is the one common layer in which the examples of a minibatch interact. It standardizes each feature using the mean and variance of that feature over the batch, then rescales and shifts with learned γ and β. Because the statistics are computed from the batch, every example's output depends on every other example's input, and the backward pass has to follow those paths. These ten problems derive the gradient of every piece, assemble the compact input gradient that most implementations use, compare it with layer norm, and then look at the cases where it changes character: inference with running statistics, a batch of one, and a feature that is constant over the batch. The five mistakes at the end are the ones that still produce a plausible gradient: the statistics taken over the wrong axis, the batch statistics held constant, an unbiased variance differentiated against a biased forward pass, γ's gradient left unsummed, and the running variance used where the batch variance belongs.

- The conventions are those of the previous pages: a gradient has the shape of the variable it is taken with respect to, $\odot$ is the elementwise product, $\mathbf{1}$ is the all-ones vector, and rows are examples. As on the previous page, dA is the code-style name for $\nabla_A L$, with the shape of A.
- The input is $X \in \mathbb{R}^{N \times d}$: N examples, d features. Batch norm works down the columns. Feature j has batch mean $\mu_j = \frac{1}{N}\sum_n X_{nj}$ and biased variance $\sigma_j^2 = \frac{1}{N}\sum_n (X_{nj} - \mu_j)^2$, and $\mu, \sigma^2 \in \mathbb{R}^d$ collect them.
- $\epsilon > 0$ is a small constant inside the square root. D is the $d \times d$ diagonal matrix with $D_{jj} = (\sigma_j^2 + \epsilon)^{-1/2}$, so the normalized input is $\hat{X} = (X - \mathbf{1}\mu^\top)D$, with entries $\hat{X}_{nj} = (X_{nj} - \mu_j)/\sqrt{\sigma_j^2 + \epsilon}$.
- The output is $Y = \hat{X}\,\mathrm{diag}(\gamma) + \mathbf{1}\beta^\top$, that is $Y_{nj} = \gamma_j \hat{X}_{nj} + \beta_j$, with learned $\gamma, \beta \in \mathbb{R}^d$. L is a scalar loss that depends on X, γ and β only through Y, and dY is given.
- One feature at a time: $x \in \mathbb{R}^N$ is column j of X, μ and σ^2 are then the scalars μ_j and σ_j^2, $\hat{x} = (x - \mu\mathbf{1})/\sqrt{\sigma^2 + \epsilon}$ is column j of $\hat{X}$, and $d\hat{x}$, dx are column j of $d\hat{X}$, dX. Sums $\sum_n$ run over the N examples.

- Running statistics $\bar{\mu}, \bar{\sigma}^2 \in \mathbb{R}^d$ are moving averages of the batch statistics, updated outside the gradient; at inference they replace μ and σ^2, and $\bar{D}$ is D built from $\bar{\sigma}^2$.

Builds on: Batched backprop: dense layers on a minibatch, Layer norm backward

Problems

1. Give the shapes of μ, σ^2, $\hat{X}$, γ, β and Y. Which entries of X does Y_{nj} depend on? difficulty ·

2. Compute $d\gamma$, $d\beta$ and $d\hat{X}$ from dY. difficulty ·

3. For one feature $x \in \mathbb{R}^N$, compute $\nabla_x \mu$ and $\nabla_x \sigma^2$. difficulty ··

4. Treat one feature's forward pass as a graph: μ is computed from x, $\sigma^2 = \frac{1}{N}\sum_n (x_n - \mu)^2$ from x and μ, and $\hat{x}_n = (x_n - \mu)/\sqrt{\sigma^2 + \epsilon}$ from x, μ and σ^2. Given $d\hat{x}$, compute the gradients $d\sigma^2 = \partial L/\partial \sigma^2$ and $d\mu = \partial L/\partial \mu$ arriving at those two nodes. difficulty ··

5. Assemble dx for one feature from $d\hat{x}$, $d\sigma^2$ and $d\mu$, and simplify it to a form that uses only $d\hat{x}$, $\hat{x}$ and σ^2. difficulty ···

6. For one feature, show that $\mathbf{1}^\top dx = 0$ and compute $\hat{x}^\top dx$. When is dx orthogonal to $\hat{x}$? difficulty ··

7. Let $\mathrm{LN}(Z)$ be layer norm without γ and β, applied to each row of a matrix Z with that row's own mean, biased variance and ϵ. Express $\hat{X}$ and dX through LN, and say which axis each layer averages over. difficulty ··

8. At inference the layer uses the running statistics: $\hat{X} = (X - \mathbf{1}\bar{\mu}^\top)\bar{D}$. Compute dX, $d\gamma$ and $d\beta$. difficulty ··

9. Train with a batch of one ($N = 1$). Compute μ, σ^2, $\hat{X}$, Y and the gradients dX, $d\gamma$, $d\beta$. difficulty ·

10. With $N \geq 2$, one feature takes the same value on every example in the batch. Compute $\hat{x}$ and dx for that feature. What does ϵ do here, and how large is the effect?

difficulty ···

Answers

1. $\mu, \sigma^2, \gamma, \beta \in \mathbb{R}^d$; $\hat{X}, Y \in \mathbb{R}^{N \times d}$; Y_{nj} depends on every entry of column j of X and on no other column
2. $d\gamma = (dY \odot \hat{X})^\top \mathbf{1}$; $d\beta = dY^\top \mathbf{1}$; $d\hat{X} = dY\ \mathrm{diag}(\gamma)$
3. $\nabla_x \mu = \frac{1}{N}\mathbf{1}$; $\nabla_x \sigma^2 = \frac{2}{N}(x - \mu\mathbf{1})$
4. $d\sigma^2 = -\dfrac{1}{2(\sigma^2+\epsilon)} \sum_n d\hat{x}_n \hat{x}_n$; $d\mu = -\dfrac{1}{\sqrt{\sigma^2+\epsilon}} \sum_n d\hat{x}_n$
5. $dx = \dfrac{1}{N\sqrt{\sigma^2+\epsilon}} \Big(N\, d\hat{x} - \big(\sum_n d\hat{x}_n\big)\mathbf{1} - \hat{x} \sum_n d\hat{x}_n \hat{x}_n \Big)$ for each feature
6. $\mathbf{1}^\top dx = 0$; $\hat{x}^\top dx = \dfrac{1}{\sqrt{\sigma^2+\epsilon}} \dfrac{\epsilon}{\sigma^2+\epsilon} \sum_n d\hat{x}_n \hat{x}_n$, which is 0 when $\epsilon = 0$
7. $\hat{X} = \mathrm{LN}(X^\top)^\top$, and dX is the transpose of layer norm's input gradient at $X^\top$ with upstream $d\hat{X}^\top$: batch norm takes its means down each column, over the N examples (axis 0), and layer norm along each row, over the d features (axis 1)
8. $dX = dY\ \mathrm{diag}(\gamma)\bar{D}$, that is $dX_{nj} = \gamma_j\, dY_{nj} / \sqrt{\bar{\sigma}_j^2 + \epsilon}$; $d\gamma = (dY \odot \hat{X})^\top \mathbf{1}$ and $d\beta = dY^\top \mathbf{1}$ with $\hat{X} = (X - \mathbf{1}\bar{\mu}^\top)\bar{D}$
9. With $N = 1$: $\mu = X^\top$, $\sigma^2 = 0$, $\hat{X} = 0$, $Y = \beta^\top$; $dX = 0$, $d\gamma = 0$, $d\beta = dY^\top$
10. On a constant feature $\hat{x} = 0$ and $dx = \dfrac{1}{\sqrt{\epsilon}} \Big(d\hat{x} - \dfrac{1}{N}\big(\sum_n d\hat{x}_n\big)\mathbf{1} \Big)$; without ϵ the forward pass would divide 0 by 0

Worked solutions

Problem 1

> Give the shapes of μ, σ^2, $\hat{X}$, γ, β and Y. Which entries of X does Y_{nj} depend on?

$\mu = \frac{1}{N} X^\top \mathbf{1} \in \mathbb{R}^d$, and likewise $\sigma^2 \in \mathbb{R}^d$.	Each is a sum over the N examples, which leaves one number per feature; $(d \times N)(N \times 1) = d \times 1$.
$X - \mathbf{1}\mu^\top$ is $N \times d$ and subtracts μ_j from every entry of column j; multiplying on the right by the diagonal D scales column j by $(\sigma_j^2 + \epsilon)^{-1/2}$, so $\hat{X}$ is $N \times d$.	$(\mathbf{1}\mu^\top)_{nj} = \mu_j$, and $(AD)_{nj} = A_{nj} D_{jj}$ for a diagonal D.
$Y = \hat{X}\,\mathrm{diag}(\gamma) + \mathbf{1}\beta^\top$ is $N \times d$, with $\gamma, \beta \in \mathbb{R}^d$.	One scale and one shift per feature, shared by every example, as the bias of the previous page's dense layer is.
$\hat{X}_{nj}$ uses X_{nj}, μ_j and σ_j^2, and μ_j and σ_j^2 are built from all N entries of column j and nothing else.	The definitions of μ_j and σ_j^2 sum over n with j fixed.
$\mu, \sigma^2, \gamma, \beta \in \mathbb{R}^d$; $\hat{X}, Y \in \mathbb{R}^{N \times d}$; Y_{nj} **depends on every entry of column j of X and on no other column**	Features never mix and examples always do, the reverse of the dense layer on the previous page, where row n of the output used only row n of the input. So everything below can be done one column at a time, and this is the layer the previous page's Problem 9 excluded from gradient accumulation.

Problem 2

> Compute $d\gamma$, $d\beta$ and $d\hat{X}$ from dY.

$Y_{nj} = \gamma_j \hat{X}_{nj} + \beta_j$.	Index form of $Y = \hat{X}\,\mathrm{diag}(\gamma) + \mathbf{1}\beta^\top$; $\hat{X}$ does not depend on γ or β.

$\partial L/\partial \gamma_j = \sum_n dY_{nj}\hat{X}_{nj}$ and $\partial L/\partial \beta_j = \sum_n dY_{nj}$.	γ_j and β_j appear in Y_{nj} for every example n, with coefficients $\hat{X}_{nj}$ and 1, and the chain rule sums over every entry that contains them.
$\partial L/\partial \hat{X}_{nj} = \gamma_j\, dY_{nj}$.	$\hat{X}_{nj}$ appears only in Y_{nj}, with coefficient γ_j.
For a matrix M with N rows, $M^\top \mathbf{1}$ adds up its rows, and $M\,\mathrm{diag}(\gamma)$ scales column j by γ_j.	$(M^\top \mathbf{1})_j = \sum_n M_{nj}$ and $(M\,\mathrm{diag}(\gamma))_{nj} = M_{nj}\gamma_j$.
$d\gamma = (dY \odot \hat{X})^\top \mathbf{1}$; $d\beta = dY^\top \mathbf{1}$; $d\hat{X} = dY\,\mathrm{diag}(\gamma)$	Shapes d, d and $N \times d$. The sums over the batch appear because γ and β are shared by every example. From here on the input gradient needs only $d\hat{X}$, one column at a time: for feature j, $d\hat{x} = \gamma_j$ times column j of dY.

Problem 3

> For one feature $x \in \mathbb{R}^N$, compute $\nabla_x \mu$ and $\nabla_x \sigma^2$.

$\partial \mu/\partial x_m = \frac{1}{N}$ for every m.	$\mu = \frac{1}{N}\sum_n x_n$ contains x_m once, with coefficient $\frac{1}{N}$.
$\partial \sigma^2/\partial x_m = \frac{2}{N}\sum_n (x_n - \mu)\big(\partial x_n/\partial x_m - \partial \mu/\partial x_m\big)$.	Chain rule on each square $(x_n - \mu)^2$, with μ kept as the function of x that it is.
$\partial \sigma^2/\partial x_m = \frac{2}{N}(x_m - \mu) - \frac{2}{N^2}\sum_n (x_n - \mu)$.	$\partial x_n/\partial x_m$ is 1 for $n = m$ and 0 otherwise, which picks out one term; step 1 gives $\partial \mu/\partial x_m = \frac{1}{N}$, the same for every n.
$\sum_n (x_n - \mu) = \sum_n x_n - N\mu = 0$.	$N\mu = \sum_n x_n$ by the definition of μ.

$\nabla_x \mu = \frac{1}{N}\mathbf{1}$; $\nabla_x \sigma^2 = \frac{2}{N}(x - \mu\mathbf{1})$	Both $N \times 1$, the shape of x. The $\frac{1}{N}$ in $\nabla_x \sigma^2$ is there because the forward pass uses the biased variance; it is the layer-norm page's Problem 2 with the batch in place of the features.

Problem 4

> Treat one feature's forward pass as a graph: μ is computed from x, $\sigma^2 = \frac{1}{N}\sum_n (x_n - \mu)^2$ from x and μ, and $\hat{x}_n = (x_n - \mu)/\sqrt{\sigma^2 + \epsilon}$ from x, μ and σ^2. Given $d\hat{x}$, compute the gradients $d\sigma^2 = \partial L/\partial \sigma^2$ and $d\mu = \partial L/\partial \mu$ arriving at those two nodes.

σ^2 feeds only the $\hat{x}_n$, and $\partial \hat{x}_n/\partial \sigma^2 = -\frac{1}{2}(x_n - \mu)(\sigma^2 + \epsilon)^{-3/2}$.	In the graph each node's inputs are held fixed when it is varied; power rule on $(\sigma^2 + \epsilon)^{-1/2}$.
$d\sigma^2 = \sum_n d\hat{x}_n \cdot \left(-\frac{1}{2}\right)(x_n - \mu)(\sigma^2 + \epsilon)^{-3/2} = -\dfrac{1}{2(\sigma^2 + \epsilon)} \sum_n d\hat{x}_n \hat{x}_n$.	The chain rule sums over the N children of σ^2, and $(x_n - \mu)(\sigma^2 + \epsilon)^{-1/2} = \hat{x}_n$ leaves one factor $(\sigma^2 + \epsilon)^{-1}$.
μ feeds every $\hat{x}_n$, with $\partial \hat{x}_n/\partial \mu = -1/\sqrt{\sigma^2 + \epsilon}$, and feeds σ^2, with $\partial \sigma^2/\partial \mu = -\frac{2}{N}\sum_n (x_n - \mu)$.	Differentiate each child of μ with its other inputs held fixed.
$\partial \sigma^2/\partial \mu = 0$.	Deviations from the mean sum to zero (Problem 3, step 4).
$d\mu = -\dfrac{1}{\sqrt{\sigma^2 + \epsilon}} \sum_n d\hat{x}_n + d\sigma^2 \cdot 0$.	The chain rule sums over both kinds of child, the $\hat{x}_n$ and σ^2.
$d\sigma^2 = -\dfrac{1}{2(\sigma^2 + \epsilon)} \sum_n d\hat{x}_n \hat{x}_n$; $d\mu = -\dfrac{1}{\sqrt{\sigma^2 + \epsilon}} \sum_n d\hat{x}_n$	Two scalars per feature. The path from μ through σ^2 is real but carries nothing, for the same reason that $\nabla_x \sigma^2$ in Problem 3 has no μ term: μ minimizes the mean squared deviation, so a small change in it moves σ^2 only to second order.

Problem 5

> Assemble dx for one feature from $d\hat{x}$, $d\sigma^2$ and $d\mu$, and simplify it to a form that uses only $d\hat{x}$, $\hat{x}$ and σ^2.

x_n feeds $\hat{x}_n$ with $\partial \hat{x}_n / \partial x_n = 1/\sqrt{\sigma^2+\epsilon}$, feeds σ^2 with $\partial \sigma^2 / \partial x_n = \frac{2}{N}(x_n - \mu)$, and feeds μ with $\partial \mu / \partial x_n = \frac{1}{N}$.	These are the children of x_n in the graph of Problem 4, each differentiated with its other inputs held fixed; x_n reaches $\hat{x}_m$ for $m \neq n$ only through μ and σ^2.
$dx_n = \dfrac{d\hat{x}_n}{\sqrt{\sigma^2+\epsilon}} + d\sigma^2 \cdot \dfrac{2}{N}(x_n - \mu) + \dfrac{d\mu}{N}.$	The chain rule sums over the three children.
$d\sigma^2 \cdot \dfrac{2}{N}(x_n - \mu) = -\dfrac{1}{N(\sigma^2+\epsilon)}(x_n - \mu) \sum_m d\hat{x}_m \hat{x}_m = -\dfrac{1}{N\sqrt{\sigma^2+\epsilon}}\, \hat{x}_n \sum_m d\hat{x}_m \hat{x}_m.$	Problem 4 for $d\sigma^2$, with the summation index renamed m; then $(x_n - \mu)/\sqrt{\sigma^2+\epsilon} = \hat{x}_n$.
$\dfrac{d\mu}{N} = -\dfrac{1}{N\sqrt{\sigma^2+\epsilon}} \sum_m d\hat{x}_m.$	Problem 4 for $d\mu$.
$dx_n = \dfrac{1}{N\sqrt{\sigma^2+\epsilon}} \Big(N\, d\hat{x}_n - \sum_m d\hat{x}_m - \hat{x}_n \sum_m d\hat{x}_m \hat{x}_m \Big).$	Put the three terms over the common factor $1/(N\sqrt{\sigma^2+\epsilon})$; the first term becomes $N\, d\hat{x}_n$.
$dx = \dfrac{1}{N\sqrt{\sigma^2+\epsilon}} \Big(N\, d\hat{x} - \big(\sum_n d\hat{x}_n\big)\mathbf{1} - \hat{x} \sum_n d\hat{x}_n \hat{x}_n \Big)$ **for each feature**	$N \times 1$. For all features at once, $dX = \frac{1}{N}\big(N\, d\hat{X} - \mathbf{1}\mathbf{1}^\top d\hat{X} - \hat{X} \odot \mathbf{1}\mathbf{1}^\top (d\hat{X} \odot \hat{X})\big) D$, where $\mathbf{1}\mathbf{1}^\top M$ puts the column sums of M in every row. Two sums per feature and elementwise work: $O(Nd)$, with $\hat{X}$ and σ^2 saved from the forward pass.

Problem 6

> For one feature, show that $\mathbf{1}^\top dx = 0$ and compute $\hat{x}^\top dx$. When is dx orthogonal to $\hat{x}$?

$\mathbf{1}^\top \hat{x} = \sum_n (x_n - \mu)/\sqrt{\sigma^2+\epsilon} = 0.$	Deviations from the mean sum to zero, whatever ϵ is.

$\hat{x}^\top \hat{x} = \dfrac{\sum_n (x_n - \mu)^2}{\sigma^2 + \epsilon} = \dfrac{N\sigma^2}{\sigma^2 + \epsilon}.$	$\sum_n (x_n - \mu)^2 = N\sigma^2$ by the definition of the biased variance.
$\mathbf{1}^\top dx = \dfrac{1}{N\sqrt{\sigma^2 + \epsilon}} \Big(N \sum_n d\hat{x}_n - N \sum_n d\hat{x}_n - (\mathbf{1}^\top \hat{x}) \sum_n d\hat{x}_n \hat{x}_n \Big) = 0.$	Problem 5 with $\mathbf{1}^\top \mathbf{1} = N$, and step 1 for the last term.
$\hat{x}^\top dx = \dfrac{1}{N\sqrt{\sigma^2 + \epsilon}} \Big(N - \dfrac{N\sigma^2}{\sigma^2 + \epsilon} \Big) \sum_n d\hat{x}_n \hat{x}_n.$	Problem 5 again: $\hat{x}^\top d\hat{x} = \sum_n d\hat{x}_n \hat{x}_n$, the middle term vanishes by step 1, and step 2 gives the last.
$\mathbf{1}^\top dx = 0$; $\hat{x}^\top dx = \dfrac{1}{\sqrt{\sigma^2 + \epsilon}} \dfrac{\epsilon}{\sigma^2 + \epsilon} \sum_n d\hat{x}_n \hat{x}_n$, **which is** 0 **when** $\epsilon = 0$	$N - N\sigma^2/(\sigma^2 + \epsilon) = N\epsilon/(\sigma^2 + \epsilon)$. Adding a constant to one feature across the batch leaves $\hat{X}$ unchanged, and so (up to ϵ) does scaling it, so the loss cannot push either way. It is the layer-norm page's Problem 8 with columns in place of rows. One consequence: a bias added to a feature just before batch norm gets gradient $\mathbf{1}^\top dx = 0$ and never moves, which is why layers followed by batch norm usually drop their bias.

Problem 7

> Let $\mathrm{LN}(Z)$ be layer norm without γ and β, applied to each row of a matrix Z with that row's own mean, biased variance and ϵ. Express $\hat{X}$ and dX through LN, and say which axis each layer averages over.

Row j of $X^\top$ is column j of X: the N values of feature j.	Transposing swaps the roles of rows and columns.
Row j of $\mathrm{LN}(X^\top)$ is $\big((x - \mu_j \mathbf{1})/\sqrt{\sigma_j^2 + \epsilon}\big)^\top$, which is column j of $\hat{X}$ as a row.	Layer norm's row mean and biased variance, taken over the N entries of that row, are μ_j and σ_j^2.

So $\hat{X} = \mathrm{LN}(X^\top)^\top$, and L depends on X through $X^\top$, then LN, then a transpose.	Step 2 for every j.
A transpose only relabels entries, so its backward pass is a transpose: dX is the transpose of the gradient at $X^\top$, and the upstream gradient at $\mathrm{LN}(X^\top)$ is $d\hat{X}^\top$.	Each entry of X is one entry of $X^\top$ with derivative 1, and likewise for $\hat{X}$.
The layer-norm page's Problem 7 on row j of $X^\top$, with upstream $d\hat{x}^\top$ and $\gamma = \mathbf{1}$, gives $\frac{1}{\sqrt{\sigma_j^2+\epsilon}}\big(d\hat{x} - \frac{1}{N}(\sum_n d\hat{x}_n)\mathbf{1} - \hat{x}\,\frac{1}{N}\sum_n d\hat{x}_n\hat{x}_n\big)$, as a row.	Its mean is $\frac{1}{N}\sum_n$ here; factoring out $\frac{1}{N}$ gives Problem 5 exactly.
$\hat{X} = \mathrm{LN}(X^\top)^\top$, and dX is the transpose of layer norm's input gradient at $X^\top$ with upstream $d\hat{X}^\top$: batch norm takes its means down each column, over the N examples (axis 0), and layer norm along each row, over the d features (axis 1)	The algebra is the same; what changes is which entries share statistics. In layer norm row n of dX depends only on row n of $d\hat{X}$, so examples stay independent. In batch norm column j of dX depends on all of column j of $d\hat{X}$, so each example's gradient depends on the rest of the batch.

Problem 8

> At inference the layer uses the running statistics: $\hat{X} = (X - \mathbf{1}\bar{\mu}^\top)\bar{D}$. Compute dX, $d\gamma$ and $d\beta$.

$\hat{X}_{nj} = (X_{nj} - \bar{\mu}_j)/\sqrt{\bar{\sigma}_j^2 + \epsilon}$, so $\hat{X}_{nj}$ depends only on X_{nj}, with slope $(\bar{\sigma}_j^2 + \epsilon)^{-1/2}$.	$\bar{\mu}$ and $\bar{\sigma}^2$ were accumulated from earlier batches and do not depend on the current X.
$Y_{nj} = \gamma_j \hat{X}_{nj} + \beta_j$ depends only on X_{nj}, with slope $\gamma_j(\bar{\sigma}_j^2 + \epsilon)^{-1/2}$.	Step 1 and the chain rule through one scalar.

$\partial L/\partial X_{nj} = \gamma_j \, dY_{nj}/\sqrt{\bar{\sigma}_j^2 + \epsilon}$, and over all entries that is dY $\operatorname{diag}(\gamma)\bar{D}$.	X_{nj} reaches L only through Y_{nj}; right-multiplying by the diagonal matrices scales column j by γ_j and by $(\bar{\sigma}_j^2 + \epsilon)^{-1/2}$.
$d\gamma$ and $d\beta$ come from Problem 2 unchanged, with this $\hat{X}$.	Problem 2 used only $Y = \hat{X}\operatorname{diag}(\gamma) + \mathbf{1}\beta^\top$, not where $\hat{X}$ came from.
$dX = dY$ $\operatorname{diag}(\gamma)\bar{D}$, **that is** $dX_{nj} = \gamma_j \, dY_{nj}/\sqrt{\bar{\sigma}_j^2 + \epsilon}$; $d\gamma = (dY \odot \hat{X})^\top \mathbf{1}$ **and** $d\beta = dY^\top \mathbf{1}$ **with** $\hat{X} = (X - \mathbf{1}\bar{\mu}^\top)\bar{D}$	$N \times d$, d and d. With fixed statistics the layer is an elementwise affine map, $Y = X\bar{D}\operatorname{diag}(\gamma) + \mathbf{1}\big(\beta - \operatorname{diag}(\gamma)\bar{D}\bar{\mu}\big)^\top$, so examples no longer interact, its gradient has no centering terms, and it can be folded into the weights and bias of the layer before it. This is the gradient used when a network is fine-tuned with batch norm frozen.

Problem 9

> Train with a batch of one ($N = 1$). Compute μ, σ^2, $\hat{X}$, Y and the gradients dX, $d\gamma$, $d\beta$.

$\mu = X^\top$, the single row as a column, and $\sigma^2 = 0$.	Each column has one entry, which is its own mean and has zero deviation from it.
$\hat{X} = (X - \mathbf{1}\mu^\top)D = 0 \cdot D = 0$, and $D = \epsilon^{-1/2}I$ is finite.	$X - \mathbf{1}\mu^\top = X - X = 0$; with $\epsilon = 0$ every entry would be $0/0$.
$Y = 0 \cdot \operatorname{diag}(\gamma) + \mathbf{1}\beta^\top = \beta^\top$.	$\mathbf{1}$ has one entry.
Y does not depend on X or γ at all, so $dX = 0$ and $d\gamma = (dY \odot 0)^\top \mathbf{1} = 0$, while $d\beta = dY^\top \mathbf{1} = dY^\top$.	Problem 2. Problem 5 agrees: with $N = 1$ and $\hat{x} = 0$ it gives $\frac{1}{\sqrt{\epsilon}}(d\hat{x} - d\hat{x} - 0) = 0$.

With $N = 1$: $\mu = X^\top$, $\sigma^2 = 0$, $\hat{X} = 0$, $Y = \beta^\top$; $dX = 0$, $d\gamma = 0$, $d\beta = dY^\top$	The layer outputs β whatever the input, so nothing below it receives a gradient and γ never trains. Small batches degrade the same way more gently: the statistics of a few examples are noisy, and every example's gradient is entangled with them.

Problem 10

> With $N \geq 2$, one feature takes the same value on every example in the batch. Compute $\hat{x}$ and dx for that feature. What does ϵ do here, and how large is the effect?

μ equals that common value, so $x - \mu\mathbf{1} = 0$ and $\sigma^2 = 0$.	The mean of equal numbers is that number, and every deviation is zero.
$\hat{x} = 0/\sqrt{0 + \epsilon} = 0$.	$\epsilon > 0$ makes the denominator $\sqrt{\epsilon}$; with $\epsilon = 0$ it would be $0/0$.
Problem 5 holds at $\sigma^2 = 0$: $dx = \frac{1}{N\sqrt{\epsilon}}\big(N\, d\hat{x} - (\sum_n d\hat{x}_n)\mathbf{1} - 0\big)$.	Its derivation needed only $\sigma^2 + \epsilon > 0$, so the map is smooth here; the last term carries the factor $\hat{x} = 0$.
On a constant feature $\hat{x} = 0$ and $dx = \frac{1}{\sqrt{\epsilon}}\Big(d\hat{x} - \frac{1}{N}\big(\sum_n d\hat{x}_n\big)\mathbf{1}\Big)$; without ϵ the forward pass would divide 0 by 0	Distribute the $\frac{1}{N}$ over the bracket. The gain is $1/\sqrt{\epsilon}$: about 316 for $\epsilon = 10^{-5}$, so the centered upstream gradient comes back several hundred times larger, and a tiny spread in a nearly constant feature is blown up to unit scale in the forward pass. For a feature with $\sigma^2 \gg \epsilon$, ϵ changes $\hat{x}$ only by a relative $\epsilon/(2\sigma^2)$, and the $\hat{x}$ component of dx in Problem 6 is of the same small order.

Where this goes wrong

1. Taking the mean over the features instead of the batch

Array code computes a mean with an axis argument, and layer norm, the normalization layer of

every transformer, takes it along the last axis.

$X \in \mathbb{R}^{N \times d}$ with rows as examples	Right so far: the layout of Problem 1.
"Normalize using the mean over the last axis, as layer norm does."	The habit that causes the mistake: layer norm's axis carried over, when for rows-as-examples the last axis is the features.
$\mu = \frac{1}{d} X\mathbf{1} \in \mathbb{R}^{N}$, one mean per example ✗	That is layer norm (Problem 7). Batch norm needs $\mu = \frac{1}{N} X^\top \mathbf{1} \in \mathbb{R}^{d}$, one per feature. When $N = d$ the wrong μ has the right length, broadcasts against the rows without error, and subtracts example j's mean from feature j.

2. Holding the batch mean and variance constant

At inference the gradient is elementwise (Problem 8), and the training forward pass looks the same with batch statistics in place of running ones.

$d\hat{X} = dY \operatorname{diag}(\gamma)$	Right so far: Problem 2.
"Training differs from inference only in which μ and σ^2 are used, so the backward pass is Problem 8 with D in place of $\bar{D}$."	The analogy that causes the mistake: running statistics are constants, but the batch statistics are computed from the current X, so they are functions of it.
$dX = dY \operatorname{diag}(\gamma) D$ ✗	It keeps only the first term of Problem 5 and drops the paths through μ and σ^2. Its column sums are $\left(\mathbf{1}^\top d\hat{X}\right) D$, not 0 (Problem 6), so it claims that shifting a feature across the batch changes the loss, which it cannot.

3. Differentiating the unbiased variance against a biased forward pass

Statistics courses, and the default variance function of some array libraries, divide by $N - 1$, and a hand-written backward pass often re-derives $\partial\sigma^2/\partial x$ from that formula.

$dx_n = \dfrac{d\hat{x}_n}{\sqrt{\sigma^2 + \epsilon}} + d\sigma^2 \dfrac{\partial \sigma^2}{\partial x_n} + \dfrac{d\mu}{N}$	Right so far: Problem 5, step 2, with $d\sigma^2$ and $d\mu$ from Problem 4.

"The sample variance is $\frac{1}{N-1}\sum_n (x_n - \mu)^2$, so $\partial\sigma^2/\partial x_n = \frac{2}{N-1}(x_n - \mu)$."	The shortcut that causes the mistake: the derivative of the unbiased estimator, while the forward pass divided by N (Problem 3).
$dx = \frac{1}{N\sqrt{\sigma^2+\epsilon}}\Big(N\,d\hat{x} - \big(\sum_n d\hat{x}_n\big)\mathbf{1} - \frac{N}{N-1}\,\hat{x}\sum_n d\hat{x}_n\hat{x}_n\Big)$ ✗	It is the derivative of a different function from the one the forward pass computed: the $\hat{x}$ term is $N/(N-1)$ times too large. It still sums to zero, so a sum-to-zero test passes, and at batch size 256 the error is under 0.4%, small enough to slip through a loose gradient check.

4. Leaving γ's gradient unsummed over the batch

Each example contributes its own term to γ's gradient, and the array of those terms already has a familiar shape.

$\partial L/\partial\gamma_j = \sum_n dY_{nj}\hat{X}_{nj}$	Right so far: Problem 2, step 2.
"Each example's contribution is $dY \odot \hat{X}$ in its own row, so the gradient is that array."	The shortcut that causes the mistake: stopping at the per-example contributions, the way $d\hat{X}$ is per example, when γ is a parameter shared by all of them.
$d\gamma = dY \odot \hat{X}$ ✗	It is $N \times d$, not d: the sum over examples is missing, and the gradient is $(dY \odot \hat{X})^\top\mathbf{1}$ (Problem 2). In array code the update $\gamma - \eta\,dY \odot \hat{X}$, with learning rate η, broadcasts silently and turns γ into one scale per example.

5. Scaling the training gradient by the running variance

A batch-norm layer keeps $\bar{\mu}$ and $\bar{\sigma}^2$ as stored attributes, while the batch statistics of the forward pass are temporaries that hand-written code has to remember to save.

In training, $\hat{X} = (X - \mathbf{1}\mu^\top)D$ with the batch statistics μ and σ^2	Right so far: the training forward pass as defined before Problem 1.

“The layer's variance is $\bar{\sigma}^2$, so use it for the $1/\sqrt{\sigma^2+\epsilon}$ in the backward pass.”	The shortcut that causes the mistake: reading the stored statistic instead of saving the one the forward pass divided by.
$dx = \frac{1}{N\sqrt{\bar{\sigma}^2+\epsilon}}\Big(N\,d\hat{x} - \big(\sum_n d\hat{x}_n\big)\mathbf{1} - \hat{x}\sum_n d\hat{x}_n\hat{x}_n\Big)$ ✗	The factor comes from differentiating the training forward pass, which divided by $\sqrt{\sigma^2+\epsilon}$ with the batch variance (Problem 5); $\bar{\sigma}^2$ is not on the path from X to L in training at all. The answer is off by the factor $\sqrt{(\sigma^2+\epsilon)/(\bar{\sigma}^2+\epsilon)}$, close to 1 once the running average has settled, so it hides, and far from 1 early in training, when $\bar{\sigma}^2$ still holds its initial value.

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 11 of *Deep Learning Math by Hand*, a 250-problem workbook with worked solutions: eulerpad.app/workbooks

eulerpad.app/practice/batch-norm-backward