A. Variance Structure of the Rank-1 Perturbations

We hereby study how variance in the score function is captured by the full-rank weight matrix $\mathbf{W}$ parameterization versus the rank-1 $\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}}$ parameterization. We first note that around a local optimum $\mathbf{W}_*$, the score function $\sum_{n=1}^N f(\mathbf{x}_n|\mathbf{W})$ can be approximated using the Hessian $\sum_{n=1}^N \nabla^2_{\mathbf{W}} f(\mathbf{x}_n|\mathbf{W})$:

$$\sum_{n=1}^N \left(f(\mathbf{x}_n|\mathbf{W}) - f(\mathbf{x}_n|\mathbf{W}_*)\right) \approx \frac{1}{2}\sum_{n=1}^N \sum_{h=1}^H \left\langle \mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}_*)\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)\right\rangle_F.$$

We can therefore characterize variance around a local optimum via expected fluctuation in the score function, $\sum_{n=1}^N \mathbb{E}\left[f(\mathbf{x}_n|\mathbf{W}) - f(\mathbf{x}_n|\mathbf{W}_*)\right]$. We compare here the effect of the two parameterizations: $\sum_{n=1}^N \mathbb{E}_{\mathbf{W}}\left[f(\mathbf{x}_n|\mathbf{W}) - f(\mathbf{x}_n|\mathbf{W}_*)\right]$ versus $\sum_{n=1}^N \mathbb{E}_s\left[f(\mathbf{x}_n|\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}}) - f(\mathbf{x}_n|\mathbf{W}_*)\right]$.

In what follows, we take fully connected networks to demonstrate that the rank-1 parameterization can have the same local variance structure as the full-rank parameterization. We first formulate the fully connected neural network in the following recursive relation. For fully connected network of width M and depth H, the score function $f(\mathbf{x}|\mathbf{W})$ can be recursively defined as:

$$\begin{aligned}
&\mathbf{x}^{(0)} = \mathbf{x},\\
&\mathbf{x}^{(h)} = \sqrt{\frac{c_\sigma}{M}}\sigma\left(\mathbf{W}^{(h)}\mathbf{x}^{(h-1)}\right), \quad 1 \le h \le H\\
&f(\mathbf{x}|\mathbf{W}) = a^{\mathrm{T}}\mathbf{x}^{(H)}.
\end{aligned}$$

Theorem 1 (Formal). *For a fully connected network of width M and depth H learned over N data points, let $\mathbf{W}_*$ denote local minimum of $\sum_{n=1}^N f(\mathbf{x}_n|\mathbf{W})$ in the space of weight matrices. Consider both full-rank perturbation $(\mathbf{W} - \mathbf{W}_*)$ and rank-1 perturbation $\left(\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}} - \mathbf{W}_*\right)$. Assume that the full-rank perturbation has the multiplicative covariance structure that*

$$\mathbb{E}_{\mathbf{W}^h}\left[\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)_{i,j}\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)_{k,l}\right] = \mathbf{W}_{*i,j}^{(h)}\mathbf{\Sigma}_{j,k}\mathbf{W}_{*k,l}^{(h)}, \tag{4}$$

for some symmetric positive semi-definite matrix $\mathbf{\Sigma}$. Let $\mathbf{s}_*^{(h)}$ denote a column vector of ones. Then if the rank-1 perturbation has covariance $\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right\rangle^{\mathrm{T}}\right] = \mathbf{\Sigma}$,

$$\begin{aligned}
&\sum_{n=1}^N \sum_{h=1}^H \mathbb{E}_{\mathbf{W}^h}\left[\left\langle \mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}_*)\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)\right\rangle_F\right]\\
&= \sum_{n=1}^N \sum_{h=1}^H \mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right), \nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right\rangle\right].
\end{aligned} \tag{5}$$

Theorem 1 demonstrates a correspondence between the covariance structure in the perturbation of $\mathbf{W}$ and that of s. Since $\mathbf{\Sigma}$ can be any symmetric positive semi-definite matrix, we have demonstrated here that our rank-1 parameterization can efficiently encode a wide range of fluctuations in $\mathbf{W}$. In particular, it is especially suited for multiplicative noise as advertised. If the covariance of $\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)$ is proportional to $\mathbf{W}_* \otimes \mathbf{W}_*^{\mathrm{T}}$ itself, then we can simply take the covariance of $(s - \mathbf{s}_*)$ to be identity.

We devote the rest of this section to prove Theorem 1.

Proof of Theorem 1. We first state the following lemma for the fluctuations of the score function f in $\mathbf{W}$ and s spaces.

Lemma 1. *For a fully connected network of width M and depth H learned over N data points, let $\mathbf{W}_*$ denote local minimum of $\sum_{n=1}^N f(\mathbf{x}_n|\mathbf{W})$ in the space of weight matrices. Then the local fluctuations of the score function in the space*

of the weight matrix $\mathbf{W}$ *is:*

$$\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left\langle \mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\right\rangle_F\right]$$
$$= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \operatorname{trace}\Bigg(\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\mathbf{x}_n^{(h-1)}\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}\right]$$
$$\cdot \operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H}\operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\mathbf{W}^{(\mathfrak{h})}a\right)\operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}^{(h-1)}\right)\right)\Bigg). \tag{6}$$

and in the space of the low rank representation s*,*

$$\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right), \nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right)\right\rangle\right]$$
$$= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \operatorname{trace}\Bigg(\mathbf{W}_*^{(h)}\mathbb{E}\left[\operatorname{diag}\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right)\left(\mathbf{x}_n^{(h-1)}\right)\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}\operatorname{diag}\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right)\right]\left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}$$
$$\operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H}\operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\cdot\mathbf{W}^{\mathfrak{h}}a\right)\cdot\operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right)\Bigg). \tag{7}$$

For perturbations $(\mathbf{W}-\mathbf{W}_*)$ with a multiplicative structure, we can write that

$$\mathbb{E}_{\mathbf{W}^h}\left[\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)_{i,j}\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)_{k,l}\right] = \mathbf{W}_{*i,j}\boldsymbol{\Sigma}_{j,k}\mathbf{W}_{*k,l},$$

for some matrix $\boldsymbol{\Sigma}$ (in the simplest case where $\boldsymbol{\Sigma}=\epsilon\cdot\mathrm{I}$, this corresponds to the covariance of $(\mathbf{W}-\mathbf{W}_*)$ being a decomposable tensor: $\mathbb{E}_{\mathbf{W}^h}\left[\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\right]=\epsilon\cdot\mathbf{W}_*\otimes\mathbf{W}_*^{\mathrm{T}}$). In this multiplicative perturbation case, we can show that if $\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)^{\mathrm{T}}\right]=\boldsymbol{\Sigma}$, then

$$\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\mathbf{x}_n^{(h-1)}\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}\right]$$
$$= \mathbf{W}_*^{(h)}\operatorname{diag}\left(\mathbf{x}_n^{(h-1)}\right)\boldsymbol{\Sigma}\operatorname{diag}\left(\mathbf{x}_n^{(h-1)}\right)\left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}$$
$$= \mathbf{W}_*^{(h)}\operatorname{diag}\left(\mathbf{x}_n^{(h-1)}\right)\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)^{\mathrm{T}}\right]\operatorname{diag}\left(\mathbf{x}_n^{(h-1)}\right)\left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}$$
$$= \mathbf{W}_*^{(h)}\mathbb{E}_{\mathbf{s}^{(h)}}\left[\operatorname{diag}\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)\left(\mathbf{x}_n^{(h-1)}\right)\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}\operatorname{diag}\left(\mathbf{s}^{(h)}-\mathbf{s}_*\right)\right]\left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}}.$$

Plugging this result into equations 6 and 7, we know that for any n and h,
$\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left\langle \mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\right\rangle_F\right] = \mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right), \nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right)\right\rangle\right]$.
Therefore,

$$\sum_{n=1}^{N}\sum_{h=1}^{H}\mathbb{E}_{\mathbf{W}^h}\left[\left\langle \mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}_*)\left(\mathbf{W}^{(h)}-\mathbf{W}_*^{(h)}\right)\right\rangle_F\right]$$
$$= \sum_{n=1}^{N}\sum_{h=1}^{H}\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right), \nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\left(\mathbf{s}^{(h)}-\mathbf{s}_*^{(h)}\right)\right\rangle\right]. \tag{8}$$

□

Proof of Lemma 1. We first analyze the local geometric structures of the score function in the space of the full-rank weight matrix $\mathbf{W}$ and the low rank vector s, respectively. We then leverage this Hessian information to finish our proof.

Local Geometry of the score function $f(\mathbf{x}_n|\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}})$: We can first compute the gradient of weight $\mathbf{W}$ at h-th layer for the predictive score function f of an H layer fully connected neural network taken at data point $\mathbf{x}_n$:

$$
\begin{aligned}
&\nabla_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}) \\
&= \frac{\partial \mathbf{x}_n^{(h)}}{\partial \mathbf{W}^{(h)}} \nabla_{\mathbf{x}_n^{(h)}} f(\mathbf{x}|\mathbf{W}) \\
&= \sqrt{\frac{c_\sigma}{M}} \mathrm{diag}\left(\sigma'\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right) \cdot \frac{\partial}{\partial \mathbf{x}_n^{(h)}} f(\mathbf{x}_n|\mathbf{W}) \cdot \left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathrm{diag}\left(\sigma'\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right) \cdot \prod_{\mathfrak{h}=h+1}^{H} \mathrm{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}_n^{(\mathfrak{h}-1)}\right)\right) \cdot \mathbf{W}^{\mathfrak{h}} a \cdot \left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \underbrace{\sigma'\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right) \prod_{\mathfrak{h}=h+1}^{H} \circ \sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}_n^{(\mathfrak{h}-1)}\right) \cdot \mathbf{W}^{\mathfrak{h}} a}_{v_n^{(h)}} \cdot \left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}.
\end{aligned}
$$

If we instead take the gradient over the vector s, we obtain that

$$
\begin{aligned}
&\nabla_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}}) \\
&= \left\langle \frac{\partial}{\partial \mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}), \frac{\partial \mathbf{W}^{(h)}}{\partial \mathbf{s}^{(h)}} \right\rangle_F \\
&= \left(\frac{\partial}{\partial \mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W})\right)^{\mathrm{T}} \circ \left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \mathbf{r}^{(h)} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \circ \mathbf{x}_n^{(h-1)} \cdot \left(v_n^{(h)}\right)^{\mathrm{T}} \mathbf{r}^{(h)} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \left(\mathbf{r}^{(h)} \circ v_n^{(h)}\right) \circ \mathbf{x}_n^{(h-1)} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathrm{diag}\left(\mathbf{x}_n^{(h-1)}\right) \left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \mathrm{diag}\left(\mathbf{r}^{(h)}\right) v_n^{(h)}.
\end{aligned}
$$

We can further analyze the Hessian of f:

$$
\begin{aligned}
&\nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}) \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathrm{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \mathrm{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}_n^{(\mathfrak{h}-1)}\right)\right) \mathbf{W}^{\mathfrak{h}} a\right) \mathrm{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right) \otimes \mathbf{x}_n^{(h-1)} \left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}. \quad (9)
\end{aligned}
$$

Whereas for s,

$$
\begin{aligned}
&\nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W}_* \circ \mathbf{r}\mathbf{s}^{\mathrm{T}}) \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathrm{diag}\left(\mathbf{x}_n^{(h-1)}\right) \left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \mathrm{diag}\left(\mathbf{r}^{(h)}\right) \mathrm{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \mathrm{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right) \cdot \mathbf{W}^{(\mathfrak{h})} a\right) \\
&\cdot \mathrm{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right) \mathrm{diag}\left(\mathbf{r}^{(h)}\right) \mathbf{W}_*^{(h)} \mathrm{diag}\left(\mathbf{x}_n^{(h-1)}\right). \quad (10)
\end{aligned}
$$

Variance Structures in the Score Function: Applying the results in equations 9 and 10, we obtain that

$$
\begin{aligned}
&\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left\langle \mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}, \nabla^2_{\mathbf{W}^{(h)}} f(\mathbf{x}_n|\mathbf{W}) \left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)\right\rangle_F\right] \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathbb{E}_{\mathbf{W}^{(h)}}\Bigg[\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}} \left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \\
&\operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\mathbf{W}^{\mathfrak{h}}a\right) \operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}^{(h-1)}\right)\right)\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)\mathbf{x}_n^{(h-1)}\Bigg] \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \operatorname{trace}\Bigg(\mathbb{E}_{\mathbf{W}^{(h)}}\left[\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)\mathbf{x}_n^{(h-1)}\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}}\left(\mathbf{W}^{(h)} - \mathbf{W}_*^{(h)}\right)^{\mathrm{T}}\right] \\
&\cdot \operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\mathbf{W}^{\mathfrak{h}}a\right) \operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}^{(h-1)}\right)\right)\Bigg).
\end{aligned}
$$

and that

$$
\begin{aligned}
&\mathbb{E}_{\mathbf{s}^{(h)}}\left[\left\langle \left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right), \nabla^2_{\mathbf{s}^{(h)}} f(\mathbf{x}_n|\mathbf{W}) \left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right\rangle\right] \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \mathbb{E}\left(\mathbf{W}_*^{(h)}\left(\mathbf{x}_n^{(h-1)} \circ \left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right) \circ \mathbf{r}_*^{(h)}\right)^{\mathrm{T}} \operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\cdot \mathbf{W}^{\mathfrak{h}}a\right) \\
&\cdot \operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right)\cdot \mathbf{W}_*^{(h)}\left(\mathbf{x}_n^{(h-1)} \circ \left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right) \circ \mathbf{r}_*^{(h)} \\
&= \left(\frac{c_\sigma}{M}\right)^{\frac{H-h+1}{2}} \operatorname{trace}\Bigg(\mathbf{W}_*^{(h)}\mathbb{E}\left[\operatorname{diag}\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\left(\mathbf{x}_n^{(h-1)}\right)\left(\mathbf{x}_n^{(h-1)}\right)^{\mathrm{T}} \operatorname{diag}\left(\mathbf{s}^{(h)} - \mathbf{s}_*^{(h)}\right)\right]\left(\mathbf{W}_*^{(h)}\right)^{\mathrm{T}} \\
&\operatorname{diag}\left(\prod_{\mathfrak{h}=h+1}^{H} \operatorname{diag}\left(\sigma'\left(\mathbf{W}^{(\mathfrak{h})}\mathbf{x}^{(\mathfrak{h}-1)}\right)\right)\cdot \mathbf{W}^{\mathfrak{h}}a\right) \cdot \operatorname{diag}\left(\sigma''\left(\mathbf{W}^{(h)}\mathbf{x}_n^{(h-1)}\right)\right)\Bigg).
\end{aligned}
$$

□

B. Additional Experimental Details and Hyperparameters

We experiment with both mixture of Gaussian and mixture of Cauchy priors (and variational posteriors) for the rank-1 factors. All reported results are averages over 10 runs for the image classification tasks and 25 runs for the EHR task. For Gaussian distributions in the image tasks, we achieve superior metric performance using only 1 Monte Carlo sample for each of 4 components to estimate the integral in Equation 2 for both training and evaluation, unlike much of the BNN literature, and we show further gains from using larger numbers of samples (4 and 25; see appendix C.3). For Cauchy distributions on those image tasks, we use 1 Monte Carlo sample for each of 4 components for training, and use 4 samples per component during evaluation. For the EHR task, we also use only 1 sample during training, but use 25 samples during evaluation (down from 200 samples for the Bayesian models in Dusenberry et al. (2019)). See Appendix B for details on hyperparameters. Our code uses TensorFlow and Edward2's Bayesian Layers (Tran et al., 2018); all experiments are available at `https://github.com/google/edward2`.

For rank-1 BNNs, there are three hyperparameters in addition to the deterministic baseline's: the number of mixture components (we fix it at 4); prior standard deviation (we vary among 0.05, 0.1, and 1); and the mean initialization for variational posteriors (either random sign flips with probability random_sign_init or a random normal with mean 1 and standard deviation random_sign_init). All hyperparameters for our rank-1 BNNs can be found in Tables 5, 6, and 7.

Following Section 3's ablations, we always (with one exception) use a prior with mean at 1, the average per-component log-likelihood, and initialize variational posterior standard deviations under the dropout parameterization as 10^{-3} for Gaussian priors and 10. The one exception is the Cauchy rank-1 Bayesian RNN on MIMIC-III, where we use a prior with mean 0.5.

Rank-1 BNNs apply rank-1 factors to all layers in the network except for normalization layers and the embedding layers in the MIMIC-III models. We are not Bayesian about the biases, but we do not find it made a difference.

We use a linear KL annealing schedule for 2/3 of the total number of training epochs (we also tried 1/3 and 1/4 and did not find the setting sensitive). Rank-1 BNNs use 250 training epochs for CIFAR-10/100 (deterministic uses 200); 135 epochs for ImageNet (deterministic uses 90); and 12000 to 25000 steps for MIMIC-III.

All methods use the largest batch size before we see a generalization gap in any method. For ImageNet, this is 32 TPUv2 cores with a per-core batch size of 128; for CIFAR-10/100, this is 8 TPUv2 cores with a per-core batch size of 64; for MIMIC-III this differs depending on the architecture. All CIFAR-10/100 and ImageNet methods use SGD with momentum with the same step-wise learning rate decay schedule, built on the deterministic baseline. For MIMIC-III, we use Adam (Kingma & Ba, 2014) with no decay schedule.

For MIMIC-III, all hyperparameters for the baselines match those of Dusenberry et al. (2019), except we used a batch size of 128 for the deterministic and Bayesian Embeddings models. Since Dusenberry et al. (2019) tuned each model separately, including the architecture sizes, we also tuned our rank-1 Bayesian RNN architecture sizes (for performance and memory constraints). Of note, the Gaussian rank-1 RNN has a slightly smaller architecture (rnn_dim=512 vs. 1024).

Dataset	CIFAR-10		CIFAR-100	
ensemble_size	4			
base_learning_rate	0.1			
prior_mean	1.0			
per_core_batch_size	64			
num_cores	8			
lr_decay_ratio	0.2			
train_epochs	250			
lr_decay_epochs	[80, 160, 180]			
kl_annealing_epochs	200			
l2	0.0001		0.0003	
Method	Normal	Cauchy	Normal	Cauchy
alpha_initializer	trainable_normal	trainable_cauchy	trainable_normal	trainable_cauchy
alpha_regularizer	normal_kl_divergence	cauchy_kl_divergence	normal_kl_divergence	cauchy_kl_divergence
gamma_initializer	trainable_normal	trainable_cauchy	trainable_normal	trainable_cauchy
gamma_regularizer	normal_kl_divergence	cauchy_kl_divergence	normal_kl_divergence	cauchy_kl_divergence
prior_stddev	0.1	0.1	0.1	0.01
dropout_rate (init)	0.001	10^{-6}	0.001	10^{-6}
random_sign_init	-0.5	-0.5	-1.0	-1.0

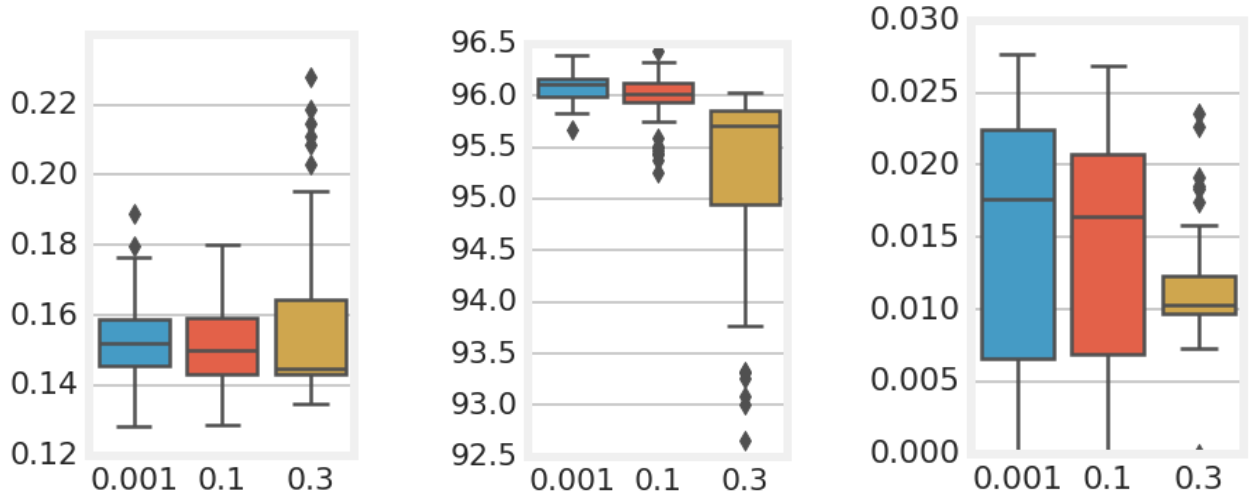
Table 5: Hyperparameter values for Rank-1 BNNs with Wide ResNet-28-10 on CIFAR-10 and CIFAR-100. Alpha and Gamma refer to the r and s vectors in the main text. The initializer determines the form of the variational posterior whereas the regularizer dictates the choice of priors. Note that all priors and approximate posteriors are mixtures.

Dataset	ImageNet	
ensemble_size	4	
base_learning_rate	0.1	
prior_mean	1.0	
per_core_batch_size	128	
num_cores	32	
lr_decay_ratio	0.1	
train_epochs	135	
lr_decay_epochs	[45, 90, 120]	
kl_annealing_epochs	90	
l2	0.0001	
Method	Normal	Cauchy
alpha_initializer	trainable_normal	trainable_cauchy
alpha_regularizer	normal_kl_divergence	cauchy_kl_divergence
gamma_initializer	trainable_normal	trainable_cauchy
gamma_regularizer	normal_kl_divergence	cauchy_kl_divergence
prior_stddev	0.05	0.005
dropout_rate (init)	0.001	10^{-6}
random_sign_init	-0.75	-0.5

Table 6: Hyperparameter values for Rank-1 BNNs with ResNet-50 on ImageNet.

Dataset	MIMIC-III	
ensemble_size	4	
embeddings_initializer	trainable_normal	
embeddings_regularizer	normal_kl_divergence	
random_sign_init	0.5	
rnn_dim	512	
hidden_layer_dim	512	
l2	1e−4	
bagging_time_precision	86400	
num_ece_bins	15	
Method	Normal	Cauchy
alpha_initializer	trainable_normal	trainable_cauchy
alpha_regularizer	normal_kl_divergence	cauchy_kl_divergence
gamma_initializer	trainable_normal	trainable_cauchy
gamma_regularizer	normal_kl_divergence	cauchy_kl_divergence
prior_mean	1.	0.5
prior_stddev	0.1	0.0001
dropout_rate (init)	0.001	5e−7
dense_embedding_dimension	32	16
embedding_dimension_multiplier	0.85827	0.984215
batch_size	128	32
learning_rate	0.00030352	0.001
fast_weight_lr_multiplier	1.	0.575
kl_annealing_steps	20000	694216
max_steps	25000	12000
bagging_aggregate_older_than	−1	$60 * 60 * 24 * 90$
clip_norm	7.29199	1.83987

Table 7: Hyperparameter values for Rank-1 Bayesian RNNs on MIMIC-III.



(a) Test NLL (b) Test Accuracy (c) Test ECE

Figure 8: Dropout-parameterized initialization for the variational distribution's standard deviations. Each boxplot is over 96 runs from a hyperparameter sweep. Using a dropout rate (and therefore standard deviation) close to zero gets much better accuracy at a slight cost of calibration error.

C. Further Ablation Studies

C.1. Initialization

There are two sets of parameters to initialize: the set of weights $\mathbf{W}$ and the variational parameters of the rank-1 distributions $q(\mathbf{r})$ and $q(\mathbf{s})$. The weights are initialized just as in deterministic networks. For the variational posterior distributions, we initialize the mean following BatchEnsemble: random sign flips of ± 1 or a draw from a normal centered at 1. This encourages each sampled vector to be roughly orthogonal from one another (thus inducing different directions for diverse solutions as one takes gradient steps); unit mean encourages the identity.

For the variational standard deviation parameters σ, we explore two approaches (Figure 8). The first is a "deterministic initialization," where σ is set close to zero such that—when combined with KL annealing—the initial optimization trajectory resembles a deterministic network's. This is commonly used for variational inference (e.g., Kucukelbir et al. (2017)). Though this aids optimization and aims to prevent underfitting, one potential reason for why BNNs still underperform is that a deterministic initialization encourages poorly estimated uncertainties: the distribution of weights may be less prone to expand as the annealed KL penalizes deviations from the prior (the cost tradeoff under the likelihood may be too high). Alternatively, we also try a "dropout initialization", where standard deviations are reparameterized with a dropout rate: $\sigma = \sqrt{p/(1-p)}$ where p is the binary dropout probability.[3] Dropout rates between 0.1 and 0.3 (common in modern architectures) imply a standard deviation of 0.3-0.65. Figure 8 shows accuracy and calibration both decrease as a function of initialized dropout rate; NLL stays roughly the same. We recommend deterministic initialization as the accuracy gains justify the minor cost in calibration.

C.2. Real-valued Scale Parameterization

As shown in Equation 3, the hierarchical prior over $\mathbf{r}$ and $\mathbf{s}$ induces a prior over the scale parameters of the layer's weights. A natural question that arises is: should the $\mathbf{r}$ and $\mathbf{s}$ priors be constrained to be positive-valued, or left unconstrained as real-valued priors? Intuitively, real-valued priors are preferable because they can modulate the sign of the layer's inputs and outputs. To determine whether this is beneficial and necessary, we perform an ablation under our CIFAR-10 setup (Section 4). In this experiment, we compare a global mixture of Gaussians for the real-valued prior, and a global mixture of log-Gaussian distributions for the positive-valued prior. For each, we tune over the initialization of the prior's standard

[3] To derive this, observe that dropout's Bernoulli noise, which takes the value 0 with probability p and $1/(1-p)$ otherwise, has mean 1 and variance $p/(1-p)$ (Srivastava et al., 2014).

deviation, and the L2 regularization for the point-wise estimated $\mathbf{W}$. For the Gaussians, we also tune over the initialization of the prior's mean.

Figure 9 displays our findings. Similar to study of priors over $\mathbf{r}$, $\mathbf{s}$, or both, we compare results across NLL, accuracy, and ECE on the test set and CIFAR-10-C corruptions dataset. We find that both setups are comparable on test accuracy, and that the real-valued setup outperforms the other on test NLL and ECE. For the corruptions task, the two setups compare equally on NLL, and differ on accuracy and ECE.

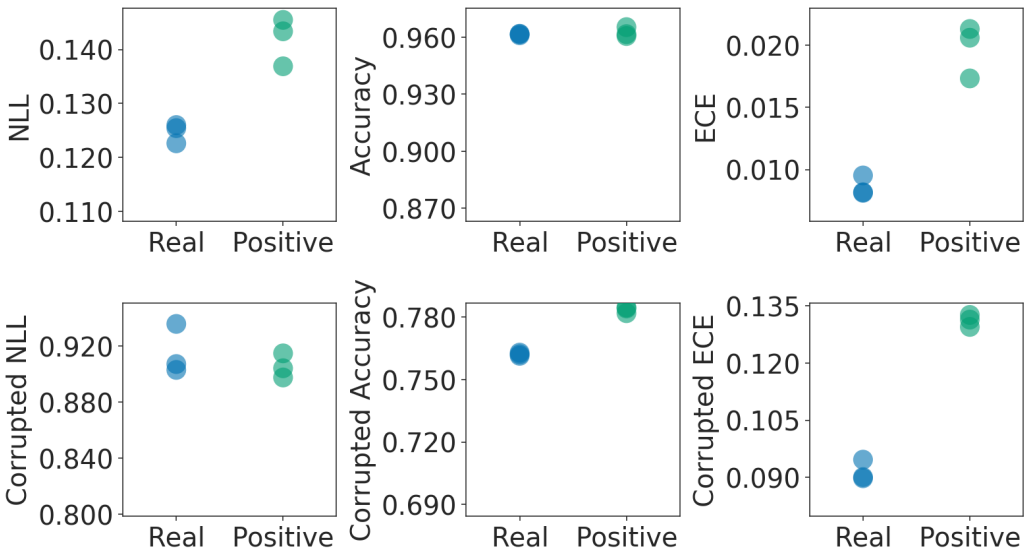


Figure 9: Real-valued vs positive-valued priors over $\mathbf{s}$ and $\mathbf{r}$, each evaluated over three runs on the CIFAR-10 test set and CIFAR-10-C corrupted dataset.

C.3. Number of Evaluation Samples

In Table 8, we experiment with using multiple weight samples, per mixture component, per example, at evaluation time for our Wide ResNet-28-10 model trained on CIFAR-10. In all cases, we use the same model that was trained using only a single weight sample (per mixture component, per example). As expected, an increased number of samples improves metric performance, with a significant improvement across all corrupted metrics. This demonstrates one of the benefits to incorporating local distributions over each mixture component, namely that given an increased computational budget, one can improve upon the metric performance at prediction time.

D. Additional Discussion and Future Directions

For future work, we'd like to push further on our results by scaling to larger ImageNet models to achieve state-of-the-art in test accuracy alongside other metrics. Although we focus on variational inference in this paper, applying this parameterization in MCMC is a promising parameter-efficient strategy for scalable BNNs. As an alternative to using mixtures trained with the average per-component log-likelihood, one can use multiple independent chains over the rank-1 factors. Another direction for future work is the straightforward extension to higher rank factors. However, prior work (Swiatkowski et al., 2019; Izmailov et al., 2019) has demonstrated diminishing returns that practically stop at ranks 3 or 5.

One surprising finding in our experimental results is that heavy-tailed priors, on a low-dimensional subspace, can significantly improve robustness and uncertainty calibration while maintaining or improving accuracy. This is likely due to the heavier tails allowing for more points in loss landscape valleys to be covered, whereas a mixture of lighter tails could place multiple modes that are nearly identical. However, with deeper or recurrent architectures, samples from the heavy-tailed posteriors seem to affect the stability of the training dynamics, leading to slightly worse predictive performance. One additional direction for future work is to explore ways to stabilize automatic differentiation through such approximate posteriors or to pair heavy-tailed priors with sub-Gaussian posteriors.

Method		NLL(↓)	Accuracy(↑)	ECE(↓)	cNLL / cA / cECE
Rank-1 BNN - Gaussian	1 sample	0.128	**96.3**	0.008	0.84 / 76.7 / 0.080
	4 samples	0.126	**96.3**	0.008	0.80 / 77.3 / 0.074
	25 samples	**0.125**	**96.3**	**0.007**	**0.77 / 77.8 / 0.070**
Rank-1 BNN - Cauchy	4 samples	**0.120**	**96.5**	0.009	**0.74 / 80.5** / 0.090
Deep Ensembles	WRN-28-5	0.115	96.3	0.008	0.84 / 77.2 / 0.089
	WRN-28-10	**0.114**	**96.6**	0.010	0.81 / 77.9 / 0.087

Table 8: Results across multiple weight samples (per mixture component, per example) at evaluation time for Wide ResNet-28-10 on CIFAR-10. Greater than 1 sample with Gaussian distributions yields a marginal improvement on in-distribution NLL and ECE, while yielding a significant improvement on all corrupted metrics. Cauchy rank-1 BNNs with 4 weight samples outperform Gaussians on all metrics except ECE. Note that training still uses a single weight sample (per mixture component, per example) for both Gaussian and Cauchy rank-1 BNNs. We include the deep ensembles results again to show that with an increased number of samples, a rank-1 WRN-28-10 can exceed an ensemble of WRN-28-5 models, which collectively have a comparable parameter count.

E. Choices of Loss Functions

E.1. Definitions

$$
\begin{aligned}
\mathbf{x} \in \mathbb{R}^d, \quad \mathbf{y}_c \in \{0, 1\}, \sum_{c=1}^{C} \mathbf{y}_c = 1 \\
\textbf{logits} &= f(\mathbf{x}, \boldsymbol{\theta}) \\
\textbf{probs} &= \mathrm{softmax}(\textbf{logits}) \\
\mathrm{softmax}(\boldsymbol{\lambda}) &= \frac{e^{\boldsymbol{\lambda}}}{\sum_{i=1}^{\|\boldsymbol{\lambda}\|} e^{\boldsymbol{\lambda}_i}} \\
p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) &= \mathrm{Categorical}(\mathbf{y}; \textbf{probs}) \\
&= \prod_{c=1}^{C} (\mathrm{softmax}(f(\mathbf{x}, \boldsymbol{\theta}))_c)^{\mathbf{y}_c} \\
-\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) &= -\sum_{c=1}^{C} \mathbf{y}_c \log \mathrm{softmax}(f(\mathbf{x}, \boldsymbol{\theta}))_c \\
&= -\mathbf{y}^\top \log \mathrm{softmax}(f(\mathbf{x}, \boldsymbol{\theta})) \\
M &= \text{num\_weight\_samples} \\
C &= \text{num\_classes}
\end{aligned}
$$

E.2. Negative log-likelihood of marginalized *logits*

$$
\begin{aligned}
&= -\mathbf{y}^\top \log \mathrm{softmax} \left(\int f(\mathbf{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \right) \\
&\approx -\mathbf{y}^\top \log \mathrm{softmax} \left(\frac{1}{M} \sum_{m=1}^{M} f(\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right)
\end{aligned} \tag{11}
$$

E.3. Negative log-likelihood of marginalized *probs*

$$
\begin{aligned}
&= -\mathbf{y}^\top \log \left\{ \int \text{softmax}(f(\mathbf{x}, \boldsymbol{\theta})) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \right\} \\
&\approx -\mathbf{y}^\top \log \left\{ \left(\frac{1}{M} \sum_{m=1}^{M} \text{softmax}(f(\mathbf{x}, \boldsymbol{\theta}^{(m)})) \right) \right\}
\end{aligned} \tag{12}
$$

E.4. Marginal Negative log-likelihood (i.e., average NLL or Gibbs cross-entropy)

$$
\begin{aligned}
&= \mathbb{E}_{p(\boldsymbol{\theta})}[-\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})] \\
&= \int -\log \left\{ p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) \right\} p(\boldsymbol{\theta}) d\boldsymbol{\theta} \\
&\approx \frac{1}{M} \sum_{m=1}^{M} \left\{ -\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right\}
\end{aligned} \tag{13}
$$

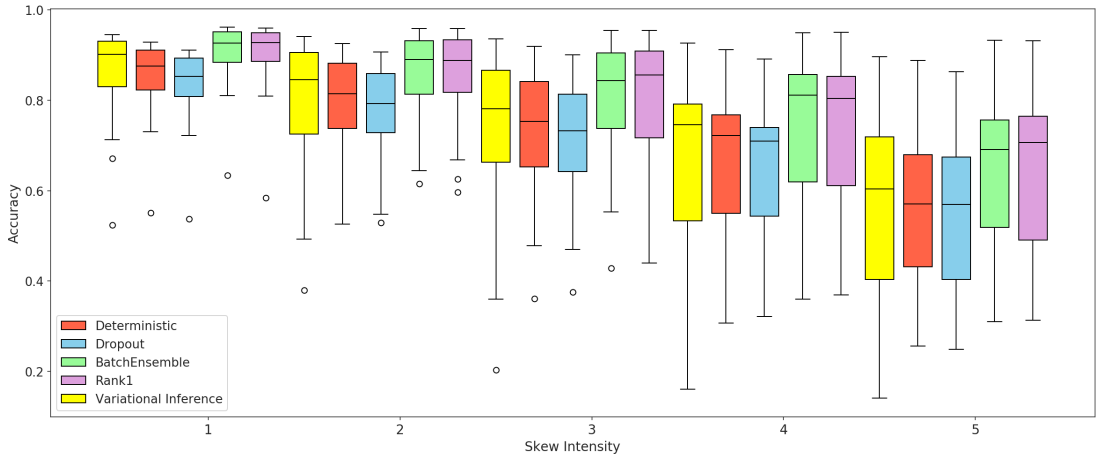
E.5. Negative log marginal likelihood (i.e., mixture NLL)

$$
\begin{aligned}
&= -\log p(\mathbf{y}|\mathbf{x}) \\
&= -\log \left\{ \int p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \right\} \\
&\approx -\log \left\{ \frac{1}{M} \sum_{m=1}^{M} p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right\} \\
&= -\log \left\{ \sum_{m=1}^{M} p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right\} + \log M \\
&= -\log \left\{ \sum_{m=1}^{M} \exp \log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right\} + \log M \\
&= -\underset{m}{\text{logsumexp}} \left\{ \log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}^{(m)}) \right\} + \log M
\end{aligned} \tag{14}
$$

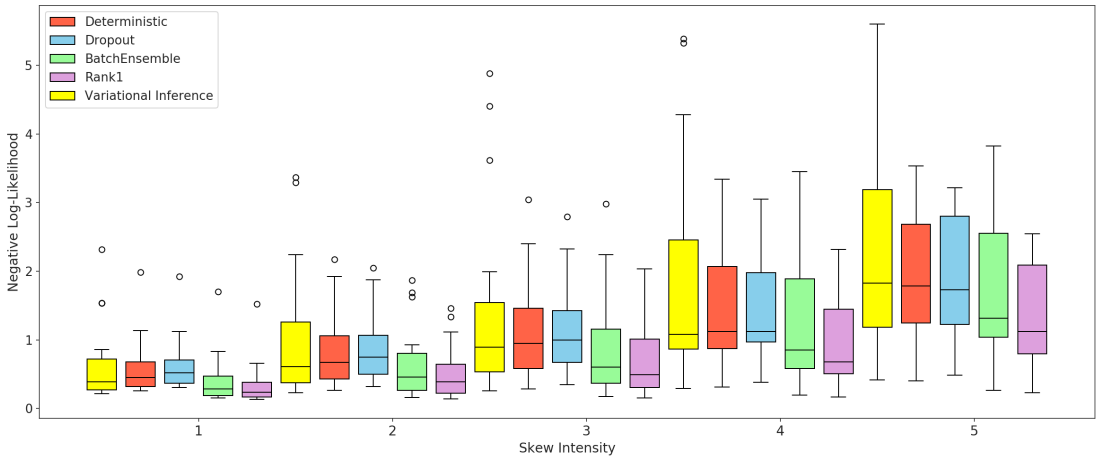
As we saw in Section 3, due to Jensen's inequality, (14) $\leq$ (13). However, we find that minimizing the upper bound (i.e. Eq. 13) to be easier while allowing for improved generalization performance. Note that for classification problems (i.e., Bernoulli or Categorical predictive distributions), Eq. 12 is equivalent to Eq. 14, though more generally, marginalizing the parameters of the predictive distribution before computing the negative log likelihood (Eq. 12) is different from marginalizing the likelihood before taking the negative log (Eq. 14), and from marginalizing the negative log likelihood (Eq. 13). Also note that though they are mathematically equivalent for classification, the formulation of Eq. 14 is more numerically stable than Eq. 12.

F. Out-of-distribution Performance

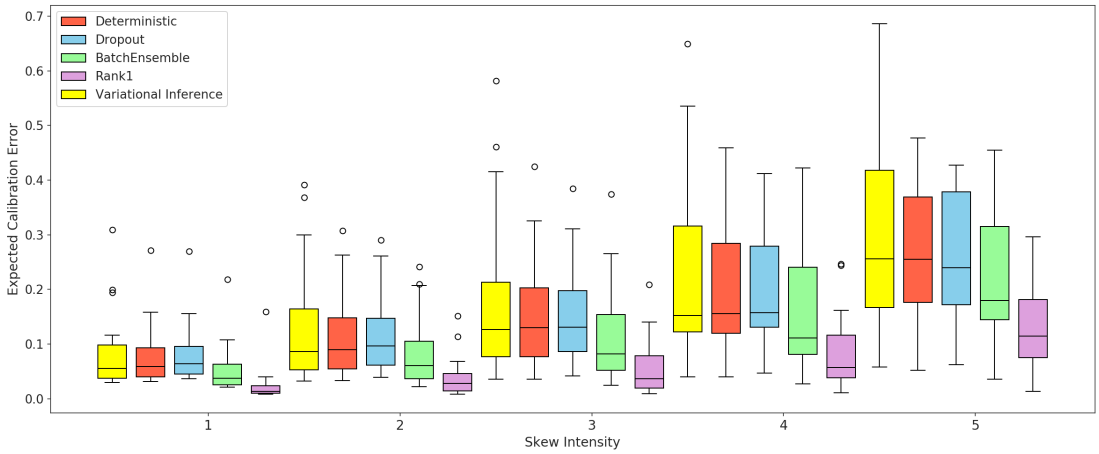
F.1. CIFAR-10-C Results



(a) Accuracy (higher is better).



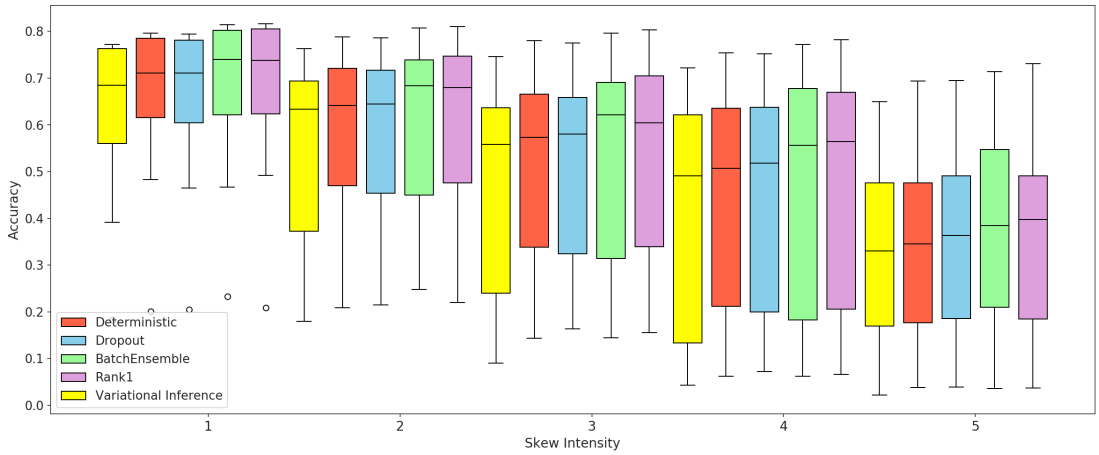
(b) Negative log-likelihood (lower is better).



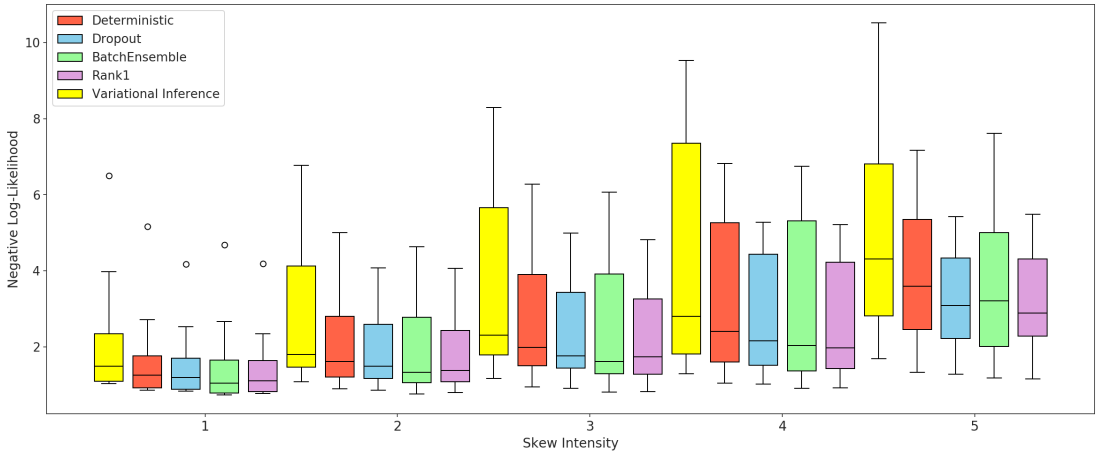
(c) Expected calibration error (lower is better).

Figure 10: Results on CIFAR-10-C showing median performance across corruption types, and for increasing settings of the skew intensity.

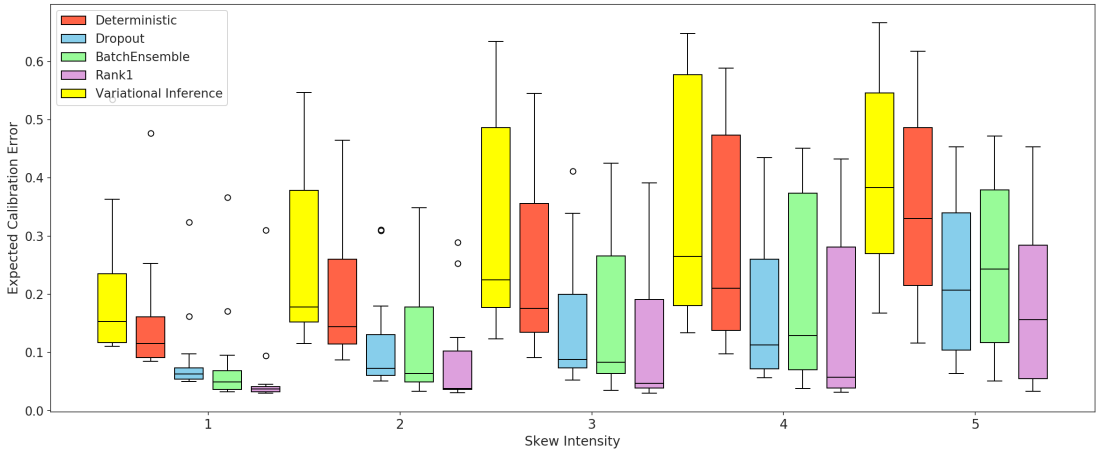
F.2. CIFAR-100-C Results



(a) Accuracy (higher is better).



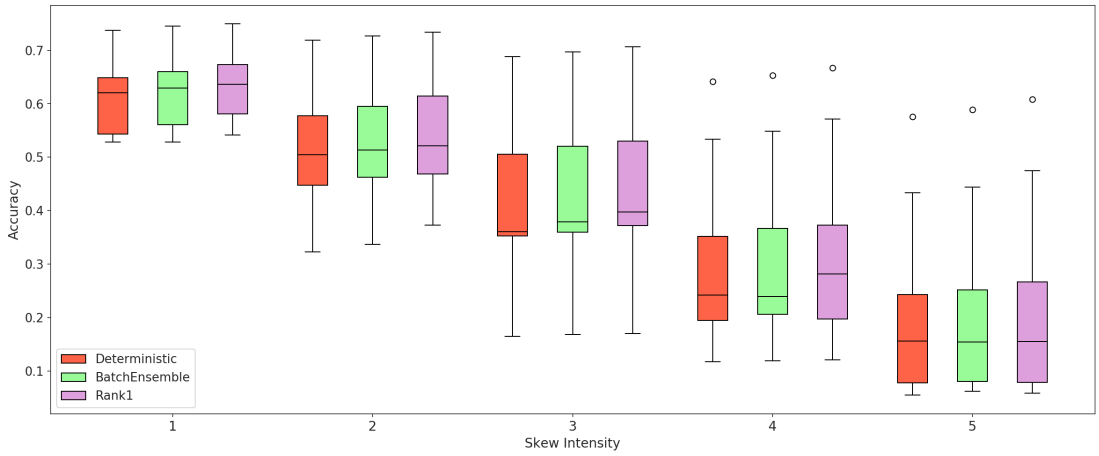
(b) Negative log-likelihood (lower is better).



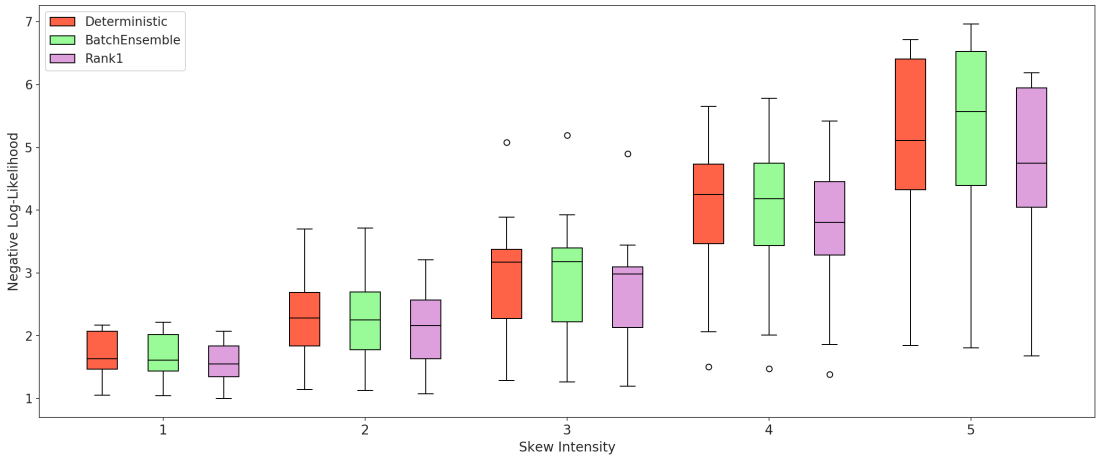
(c) Expected calibration error (lower is better).

Figure 11: Results on CIFAR-100-C showing median performance across corruption types, and for increasing settings of the skew intensity.

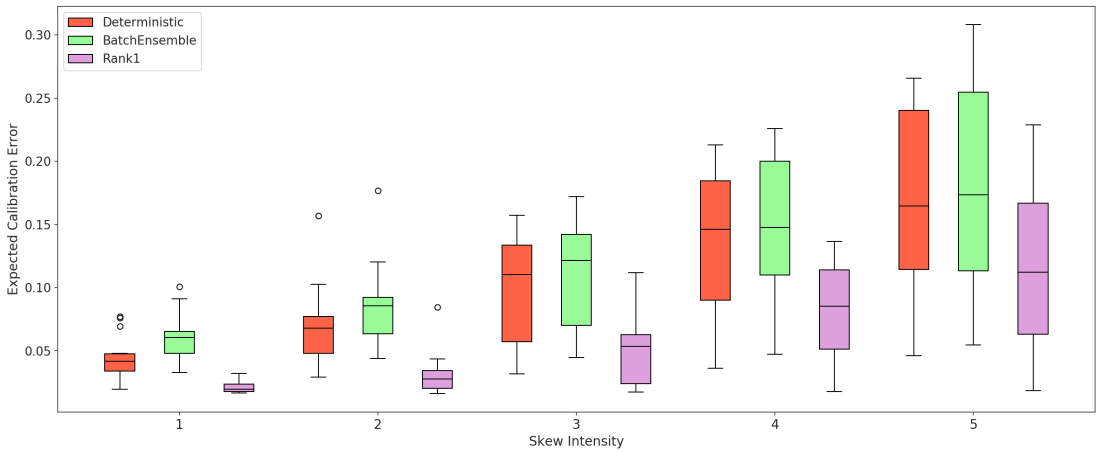
F.3. ImageNet-C Results



(a) Accuracy (higher is better).



(b) Negative log-likelihood (lower is better).



(c) Expected calibration error (lower is better).

Figure 12: Results on ImageNet-C showing median performance across corruption types, and for increasing settings of the skew intensity.