A STATISTICAL THEORY OF COLD POSTERIORS IN DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

To get Bayesian neural networks to perform comparably to standard neural networks it is usually necessary to artificially reduce uncertainty using a "tempered" or "cold" posterior. This is extremely concerning: if the generative model is accurate, Bayesian inference/decision theory is optimal, and any artificial changes to the posterior should harm performance. While this suggests that the prior may be at fault, here we argue that in fact, BNNs for image classification use the wrong likelihood. In particular, standard image benchmark datasets such as CIFAR-10 are carefully curated. We develop a generative model describing curation which gives a principled Bayesian account of cold posteriors, because the likelihood under this new generative model closely matches the tempered likelihoods used in past work.

1 INTRODUCTION

Recent work has highlighted that Bayesian neural networks (BNNs) typically have better predictive performance when we "sharpen" the posterior (Wenzel et al., 2020). In stochastic gradient Langevin dynamics (SGLD) (Welling & Teh, 2011), this can be achieved by multiplying the log-posterior by $1/T$, where the "temperature", T is smaller than 1 (Wenzel et al., 2020). Broadly the same effect can be achieved in variational inference by "tempering", i.e. downweighting the KL term. As noted in Wenzel et al. (2020), this approach has been used in many recent papers to obtain good performance, albeit without always emphasising the importance of this factor (Zhang et al., 2017; Bae et al., 2018; Osawa et al., 2019; Ashukha et al., 2020).

These results are puzzling if we take the usual Bayesian viewpoint, which says that the Bayesian posterior, used with the right prior, and in combination with Bayes decision theory should give optimal performance (Jaynes, 2003). Thus, these results may suggest we are using the wrong prior. While new priors have been suggested (e.g. Ober & Aitchison, 2020), they give only minor improvements in performance — certainly nothing like enough to close the gap to carefully trained non-Bayesian networks. In contrast, tempered posteriors directly give performance comparable to a carefully trained finite network.

The failure to develop an effective prior suggests that we should consider alternative explanations for the effectiveness of tempering. Here, we consider the possibility that it is predominantly (but not entirely) the *likelihood*, and not the prior that is at fault. In particular, we note that standard image benchmark datasets such as ImageNet and CIFAR-10 are carefully curated, and that it is important to consider this curation as part of our generative model. We develop a simplified generative model describing dataset curation which assumes that a datapoint is included in the dataset only if there is unanimous agreement on the class amongst multiple labellers. This model naturally multiplies the effect of each datapoint, and hence gives posteriors that closely match tempered or cold posteriors. We show that toy data drawn from our generative model of curation can give rise to optimal temperatures being smaller than 1. Our model predicts that cold posteriors will not be helpful when the original underlying labels from all labellers are available. While these are not available for standard datasets such as CIFAR-10, we found a good proxy: the CIFAR-10H dataset (Peterson et al., 2019), in which ~ 50 humans annotators labelled the CIFAR-10 test-set (we use these as our training set, and use the standard CIFAR-10 training set for test-data). As expected, we find strong cold-posterior effects when using the original single-label, which are almost entirely eliminated when using the 50 labels from CIFAR-10H. In addition, curation implies that each label is almost certain to be correct, which is one way to understand the statistical patterns exploited by cold posteriors. As such, if we destroy

$Y_1 = Y_2 = Y_3 = \text{cat}$ $\quad$ $Y_1 = Y_2 = Y_3 = \text{dog}$ $\quad$ $Y_1 = Y_2 = \text{cat}; \quad Y_3 = \text{dog}$

$Y = \text{cat}$ $\quad$ $Y = \text{dog}$ $\quad$ $Y = \texttt{None}$

Figure 1: A simple example of our generative process describing dataset curation, with $S = 3$. Labellers have the task of classifying images as cats or dogs. The first image is unambiguously a cat, so the three labellers agree, consensus is reached, and the image is included in the dataset. The second image is unambiguously a dog, the labellers agree, consensus is reached, and the image is included in the dataset. However, the third image is ambiguous: the labellers disagree, consensus is not reached, and the image may be excluded from the dataset.

this pattern by adding noise to the labels, the cold posterior effect should disappear. We confirmed that with increasing label noise, the cold posterior effect disappears and eventually reverses (giving better performance at temperatures close to 1).

2 Background: cold and tempered posteriors

Tempered (e.g. Zhang et al., 2017) and cold (Wenzel et al., 2020) posteriors differ slightly in how they apply the temperature parameter. For cold posteriors, we scale the whole posterior, whereas tempering is a method typically applied in variational inference, and corresponds to scaling the likelihood but not the prior,

$$\log \mathrm{P}_{\text{cold}}\left(\theta|X,Y\right) = \tfrac{1}{T}\log \mathrm{P}\left(X,Y|\theta\right) + \tfrac{1}{T}\log \mathrm{P}\left(\theta\right) + \text{const} \tag{1}$$

$$\log \mathrm{P}_{\text{tempered}}\left(\theta|X,Y\right) = \tfrac{1}{\lambda}\log \mathrm{P}\left(X,Y|\theta\right) + \quad \log \mathrm{P}\left(\theta\right) + \text{const}\,. \tag{2}$$

While cold posteriors are typically used in SGLD, tempered posteriors are usually targeted by variational methods. In particular, variational methods apply temperature scaling to the KL-divergence between the approximate posterior, $\mathrm{Q}\left(\theta\right)$ and prior,

$$\mathcal{L} = \mathbb{E}_{\mathrm{Q}(\theta)}\left[\log \mathrm{P}\left(X,Y|\theta\right)\right] - \lambda\, \mathrm{D}_{\mathrm{KL}}\left(\mathrm{Q}\left(\theta\right) || \,\mathrm{P}\left(\theta\right)\right). \tag{3}$$

Note that the only difference between cold and tempered posteriors is whether we scale the prior, and if we have Gaussian priors over the parameters (the usual case in Bayesian neural networks), this scaling can be absorbed into the prior variance,

$$\tfrac{1}{T}\log \mathrm{P}_{\text{cold}}\left(\theta\right) = -\tfrac{1}{2T\sigma^2_{\text{cold}}}\sum_i \theta_i^2 + \text{const} = -\tfrac{1}{2\sigma^2_{\text{tempered}}}\sum_i \theta_i^2 + \text{const} = \log \mathrm{P}_{\text{cold}}\left(\theta\right). \tag{4}$$

in which case, $\sigma^2_{\text{cold}} = \sigma^2_{\text{tempered}}/T$, so the tempered posteriors we discuss are equivalent to cold posteriors with rescaled prior variances.

3 Methods: a generative model for curated datasets

Standard image datasets such as CIFAR-10 and ImageNet are carefully curated to include only unambiguous examples of each class. For instance, in CIFAR-10, student labellers were paid per hour (rather than per image), were instructed that "It's worse to include one that shouldn't be included than to exclude one", and then Krizhevsky (2009) "personally verified every label submitted by the labellers". For ImageNet, Deng et al. (2009) required the consensus of a number of Amazon Mechanical Turk labellers before including an image in the dataset.

To understand the statistical patterns that might emerge in these curated datasets, we consider a highly simplified generative model of consensus-formation. In particular, we draw a random image X from

some underlying distribution over images, $\mathrm{P}(X)$, and ask S humans to assign a label, $\{Y_s\}_{s=1}^S$ (e.g. using Mechanical Turk). We force every labeller to label every image and if the image is ambiguous they are instructed to give a random label. If all the labellers agree, $Y_1 = Y_2 = \cdots = Y_S$, consensus is reached and we include the datapoint in the dataset. If any of the labellers disagree consensus is not reached, and we exclude the datapoint (Fig. 1), Formally, the observed random variable, Y, is taken to be the usual label if consensus was reached and `None` if consensus was not reached (Fig. 2B),

$$Y|\{Y_s\}_{s=1}^S = \begin{cases} Y_1 & \text{if } Y_1 = Y_2 = \cdots = Y_S \\ \texttt{None} & \text{otherwise} \end{cases} \tag{5}$$

Taking the human labels, Y_s, to come from the set $\mathcal{Y}$, so $Y_s \in \mathcal{Y}$, the consensus label, Y, could be any of the underlying labels in $\mathcal{Y}$, or `None` if no consensus is reached, so $Y \in \mathcal{Y} \cup \{\texttt{None}\}$. When consensus was reached, the likelihood is,

$$\mathrm{P}(Y = y|X, \theta) = \mathrm{P}\left(\{Y_s = y\}_{s=1}^S|X, \theta\right) = \prod_{s=1}^S \mathrm{P}(Y_s = y|X, \theta) = \mathrm{P}(Y_s = y|X, \theta)^S \tag{6}$$

where we have assumed labellers are IID. This would appear to directly give an account of tempering, as we have taken the single-labeller likelihood to the power S, which is equivalent to setting $\lambda = 1/S$. However, to see how the full generative model functions we need to go on to consider the case in which consensus was not reached,

$$\mathrm{P}(Y = \texttt{None}|X, \theta) = 1 - \sum_{y \in \mathcal{Y}} \mathrm{P}(Y = y|X, \theta) = 1 - \sum_{y \in \mathcal{Y}} \mathrm{P}(Y_s = y|X, \theta)^S . \tag{7}$$

To understand the impact of our model of consensus formation, note that the probability of a particular class-label can be separated into two terms, a probability of consensus, and the probability of a particular class given that consensus was reached,

$$\mathrm{P}(Y = y|X, \theta) = \mathrm{P}(Y = y|Y \neq \texttt{None}, X, \theta)\, \mathrm{P}(Y \neq \texttt{None}|X, \theta) . \tag{8}$$

Remarkably, knowing there was consensus gives us information about the weights, even in the absence of the class label,

$$\mathrm{P}(Y \neq \texttt{None}|X, \theta) = \sum_{y \in \mathcal{Y}} \mathrm{P}(Y_s = y|X, \theta)^S , \tag{9}$$

in essence, telling us that the output probability was close to 1 for one of the class labels, without telling us which one. Schematically (Fig. 3), we see that the datapoints with a consensus class-label, "cat" or "dog", lie far from the decision boundary where the class is unambiguous, and consensus is easily reached. In contrast, in regions close to the decision boundary the inputs are ambiguous, which tends to produce disagreement in the labellers, leading to noconsensus. Thus, the existence of one or more consensus points in a region implies that decision boundaries do not go through that region, giving us information about the decision boundary location, even if the label is not known. Concurrent work has shown that this likelihood can be used to explain classical semi-supervised likelihoods (Aitchison, 2020), so this term really does give information about the neural network parameters. Finally, the label probability, conditioned on consensus,

$$\mathrm{P}(Y = y|Y \neq \texttt{None}, X, \theta) = \frac{\mathrm{P}(Y = y|X, \theta)}{\mathrm{P}(Y \neq \texttt{None}|X, \theta)} = \frac{\mathrm{P}(Y_s = y|X, \theta)^S}{\sum_{y \in \mathcal{Y}} \mathrm{P}(Y_s = y|X, \theta)^S} \tag{10}$$

simply represents a "reparameterised" softmax.

In datasets where the noconsensus inputs are known it is clear we should use the full likelihood, (Eq. 6 and 7). The question is: in real-world datasets, where we only know the consensus inputs and the noconsensus inputs have been thrown away (Fig. 2C), can we use Eq. (10), a reparameterisation of the softmax-categorical probabilities, for the known consensus points? The answer is no because in Bayesian inference, we do not get to just pick a sensible-looking conditional probability distribution, such as Eq. 10, to use as the likelihood. Instead, we need to write down the full generative model, and marginalise over unknown latents. To write down the generative model in the case of unknown noconsensus images (Fig. 2C), we need to take X to be an unobserved latent variable, and take Z to be an observed random variable,

$$Z|X, \{Y_s\}_{s=1}^S = \begin{cases} X & \text{if } Y_1 = Y_2 = \cdots = Y_s \\ \texttt{None} & \text{otherwise} \end{cases} \tag{11}$$

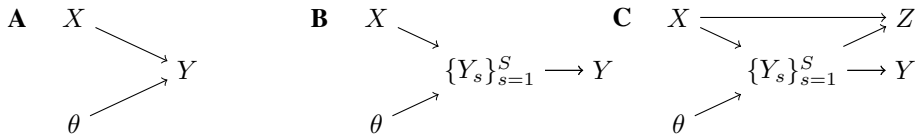

Figure 2: **A** The standard generative model for supervised tasks, assuming no data curation. **B** Generative model of data curation, where the consensus and noconsensus images are known. **C** Generative model of data curation, where the noconsensus images are unknown. The underlying images, X are no longer observed. Instead, we observe Z, where $Z = X$ if consensus is reached ($Y_1 = Y_2 = \cdots = Y_S$) and $Z = \texttt{None}$ otherwise.

which is the underlying image, X, if consensus is reached, and $\texttt{None}$ otherwise. As Z depends on θ (Fig. 2C), we cannot take the usual shortcut of using $\mathrm{P}\left(Y|Z,\theta\right)$; we must instead use the full likelihood, $\mathrm{P}\left(Y,Z|\theta\right)$,

$$\begin{aligned}\mathrm{P}\left(Y,Z|\theta\right) &= \sum_{\{Y_s\}_{s=1}^S} \int dX\, \mathrm{P}\left(X, Y, \{Y_s\}_{s=1}^S, Z|\theta\right) \\ &= \sum_{\{Y_s\}_{s=1}^S} \int dX\, \mathrm{P}\left(X\right) \left[\prod_{s=1}^S \mathrm{P}\left(Y_s|X,\theta\right)\right] \mathrm{P}\left(Y|\{Y_s\}_{s=1}^S\right) \mathrm{P}\left(Z|X,\{Y_s\}_{s=1}^S\right).\end{aligned} \tag{12}$$

for y not $\texttt{None}$,

$$\mathrm{P}\left(Y = y|\{Y_s\}_{s=1}^S\right) = \begin{cases} 1 & \text{if } Y_s = y \text{ for all } s \in \{1,\ldots,S\} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

and,

$$\mathrm{P}\left(Z|X,\{Y_s\}_{s=1}^S\right) = \begin{cases} \delta(Z - X) & \text{if } Y_1 = Y_2 = \cdots = Y_S \\ \mathbb{I}_{Z=\texttt{None}} & \text{otherwise} \end{cases} \tag{14}$$

where δ is the Dirac-delta function, and where the indicator function, $\mathbb{I}_{Z=\texttt{None}}$ is 1 if $Z = \texttt{None}$ and 0 otherwise. Substituting Eq. (13) and Eq. (14) into Eq. (12),

$$\mathrm{P}\left(Y = y, Z|\theta\right) = \mathrm{P}\left(X\right) \prod_{s=1}^S \mathrm{P}\left(Y_s = y|X,\theta\right) \propto \mathrm{P}\left(Y_s = y|X,\theta\right)^S, \tag{15}$$

where the proportionality arises because $\mathrm{P}\left(X\right)$ does not depend on the parameters of interest, θ. Note that this is proportional to Eq. (6) above, and not Eq. (10), so this does not just represent a reparameterisation of the softmax.

Finally, this explanation for the cold posterior effect would suggest that we would see cold posteriors even in maximum a-posteriori (MAP) inference, as we confirm in Appendix A. This is expected as it is "common knowledge" (but we do not know of a good reference) that while weight-decay is closely related to MAP inference with Gaussian priors, the best performing value of the weight decay coefficient tend to be lower than those suggested by untempered MAP inference.

4 Differences between cold posterior and dataset curation setups

Our model of data curation provides a direct explanation for the effectiveness of cold posteriors, as Eq. (6) takes the underlying likelihoods, $\mathrm{P}\left(Y_s|X,\theta\right)$ to the power S, which has exactly the same effect as $1/\lambda$ in tempered posteriors (Eq. 2). However, it is important to note two differences between the cold posterior setup and the ideal setup for our generative model. First, our generative model assumes that the noconsensus points are known, or if they are unknown, we can compute the integral,

$$\mathrm{P}\left(Y = \texttt{None}, Z = \texttt{None}|\theta\right) = \int dX\;\; \mathrm{P}\left(X\right) \left(1 - \sum_{y \in \mathcal{Y}} \mathrm{P}\left(Y_s = y|X,\theta\right)^S\right) \tag{16}$$

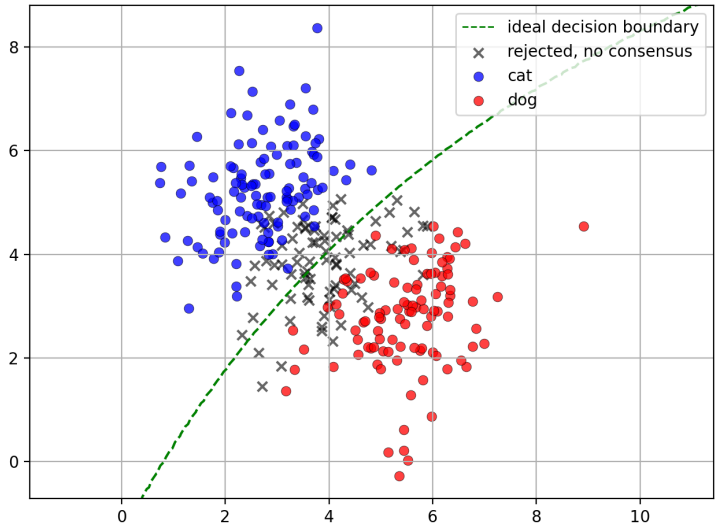

Figure 3: Illustrative example of artificial clustering induced in the dataset by rejection of ambiguous no-consensus points. The input points for "cat" and "dog" classes are generated from separate 2D Gaussian distributions, and the classifier (and decision boundary) comes from the ratio of the Gaussian probability density functions. For the consensus processes, we used $S = 7$ (we used a relatively large value to make the effects unambiguous).

which is obtained by evaluating Eq. (12) in the case where $Y = \texttt{None}$, and substituting Eq. (7). Of course, in reality it is not possible to compute this integral because we do not know the underlying $\mathrm{P}(X)$ (and usually, we do not even have samples from this distribution). In contrast, the standard cold-posterior setup entirely ignores these terms. Second, in the usual setting, the test data is also subject to the same consensus-formation process, in which case, we should use Eq. (10) for prediction. In contrast, in the standard cold-posterior setting, we use the single-labeller distribution, $\mathrm{P}(Y_s|X, \theta)$. (Note that if the test-set was drawn "from the wild", without dataset curation, and labelled by a single labeller, then we should use $\mathrm{P}(Y_s|X, \theta)$ for prediction). Overall then, while our model of data curation offers a potential explanation of the benefits of tempering, the differences in setup imply that we cannot expect $\lambda^* = 1/S$, to hold exactly, where λ^* is the optimal temperature, and this is confirmed in our results on toy data below.

5 Results

5.1 Data sampled from a known Gaussian process model

In the introduction, we took it as given that if the prior is correct, the optimal approach is to use the true Bayesian posterior, avoiding either tempering or cold posteriors. To check that this is indeed the case, we tested the performance, measured as test-log-likelihood for tempered posteriors using data generated from a known Gaussian process model. We uniformly generated 50 input points on the 1D interval [-10, 10], and used a Gaussian process with squared exponential kernel with a standard deviation of 4, and kernel bandwidth of 1. For inference, we used reparameterised VI, with an approximate posterior given by multiplying the prior by a single Gaussian factor for each datapoint (Ober & Aitchison, 2020).

In particular, we used a Gaussian process prior for the function values, $\mathbf{u}$, at the training inputs (Williams & Rasmussen, 2006),

$$\mathrm{P}(\mathbf{f}|\mathbf{x}) = \mathcal{N}(\mathbf{f}; \mathbf{0}, \mathbf{K}(\mathbf{x})) \tag{17}$$

and we use an approximate posterior (Hensman et al., 2015; Matthews et al., 2016; Ober & Aitchison, 2020) defined by multiplying the prior by a Gaussian with diagonal covariance, where we treat $\mathbf{v}$ and

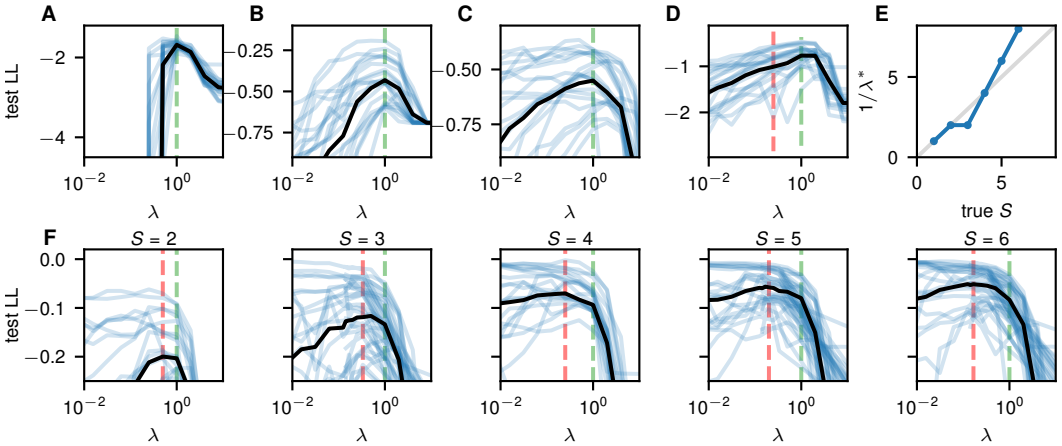

Figure 4: Tempering with data generated from a toy Gaussian process model. **A** Performance of tempered posteriors on GP regression. The green dashed line highlights $\lambda = 1$. The black line represents the mean of 20 runs (translucent blue lines). **B** GP classification. **C** GP classification with our model of consensus formation. We train and test using our exact log-likelihood, and we include knowledge of the no-consensus input datapoints in the datasets. **D** Data generated from our model of consensus formation, with $S = 4$. Training uses $\mathrm{P}\left(Y_s|X,\theta\right)$ log-likelihood (excluding noconsensus points), but testing uses the exact test-log-likelihood (Eq. 6 and 7). The red dashed line lies at $\lambda = 1/S = 1/4$. **E** Data generated from our model of consensus formation, with $S = 4$. Training and testing follow the standard cold-posterior setup, excluding noconsensus points and using $\mathrm{P}\left(Y_s|X,\theta\right)$. The x-axis gives the number of labellers in the underlying generative process, and the y-axis gives the reciprocal of the optimal temperature. **F** Plots underlying **E**. The red dashed lines indicate $\lambda = 1/S$.

$\mathbf{\Lambda}^{-1}$ as variational parameters.

$$\mathrm{Q}\left(\mathbf{f}\right) \propto \mathrm{P}\left(\mathbf{f}|\mathbf{x}\right)\mathcal{N}\left(\mathbf{v};\mathbf{f},\mathbf{\Lambda}^{-1}\right) = \mathcal{N}\left(\mathbf{f};\mathbf{\Sigma}\mathbf{\Lambda}\mathbf{v},\mathbf{\Sigma}\right) \quad \text{where} \quad \mathbf{\Sigma} = \left(\mathbf{K}^{-1}(\mathbf{x}) + \mathbf{\Lambda}\right)^{-1} \tag{18}$$

Note that this captures the true posterior in the case of regression (where $\mathbf{v}$ is set to $\mathbf{y}$ and $\mathbf{\Lambda}^{-1}$ is the output noise covariance; Ober & Aitchison, 2020). We then optimize the tempered ELBO (e.g. Zhang et al., 2017),

$$\mathcal{L} = \mathbb{E}_{\mathrm{Q}(\mathbf{f})}\left[\log \mathrm{P}\left(\mathbf{y}|\mathbf{f}\right) + \lambda\left(\log \mathrm{P}\left(\mathbf{f}|\mathbf{x}\right) - \log \mathrm{Q}\left(\mathbf{f}\right)\right)\right]. \tag{19}$$

And we used the standard GP approach for prediction at test points (Williams & Rasmussen, 2006).

Initially, we tried using standard regression and classification generative models, without including our model of consensus formation. Unsurprisingly, $\lambda = 1$, corresponding to the Bayesian posterior, is optimal for GP regression (Fig. 4A) and classification (Fig. 4B). For GP regression, we use a Gaussian likelihood with standard deviation 1, and for classification, we use a standard sigmoid probability with a Bernoulli likelihood.

Next, we confirmed that even under our new, more complex generative model of dataset curation $\lambda = 1$ was still optimal (Fig. 4C), if we trained and tested using the correct form for the log-likelihood. In particular, we considered no-consensus inputs as known and used Eq. 7 to incorporate their likelihood. However, in standard benchmarks, only the consensus inputs are known. As such, next we generated curated data, trained using $\mathrm{P}\left(Y_s|X,\theta\right)$ and ignoring noconsensus points, but tested on the correct likelihood, including noconsensus points (Fig. 4D); the optimal temperature remains around 1. Finally, we considered the standard cold/tempered posterior setup, where we use $\mathrm{P}\left(Y_s|X,\theta\right)$, excluding noconsensus points, for training and testing. The optimal λ indeed fell with S (Fig. 4EF), giving a potential explanation for the cold posterior effect. As expected, we have $S \approx 1/\lambda^*$, but the relation does not appear to hold exactly due to the mismatches discussed in Sec. 4. Finally, we show in Appendix B that the same patterns emerge in a Bayesian neural network, using Langevin sampling for inference.

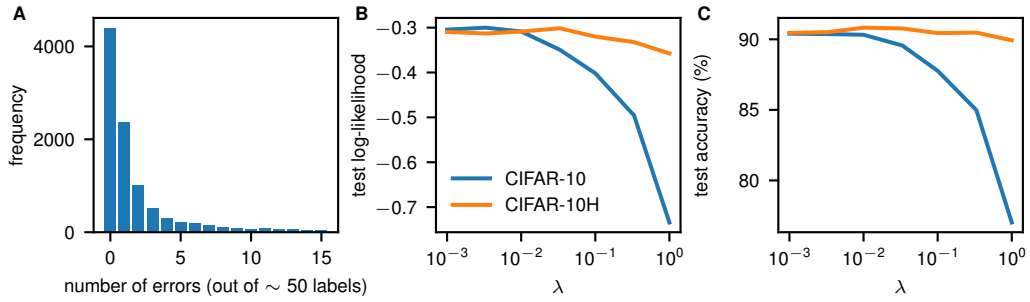

Figure 5: **A** The number of errors (defined as labellers who disagree with the most popular category) out of around 50 labels. **B** The test log-likelihood for different values of λ for training with the standard single-labels provided by CIFAR-10 and the 50 labels provided by CIFAR-10H. Note that here we have swapped the identities of the train and test set. **C** As in **B**, but for accuracy.

5.2 CIFAR-10H

One prediction made by our framework is that if we had access to the original underlying human labels on the full dataset, including images that were rejected because consensus was not reached, then tempering should not be necessary. Obviously, all this additional information is not available for standard datasets such as CIFAR-10. However, we are able to get close by considering the CIFAR-10H dataset (Peterson et al., 2019). The authors of this work asked around 50 human labellers to label the CIFAR-10 test-set. As we might expect given the careful curation that went in to creating the original CIFAR-10 dataset (Krizhevsky, 2009), for almost half of the datapoints, all ~ 50 labellers agreed, and more than three-quarters of images had 2 or fewer disagreements (corresponding to 4%) of labellers (Fig. 5A). While it is not possible to estimate S without having information about the unknown image distribution, the high level of agreement would indicate that the effective value of S is large — potentially even larger than 10.

Next, we trained a neural network on the ~ 50 labels provided by the CIFAR-10H dataset. As these labels are provided for the test-set with 10,000 images, this required us to swap the identities of the test and training sets (so that our training set consisted of 10,000 points, each with around 50 labels, and the test set consists of 50,000 points, with the single label from the original CIFAR-10 dataset). We compared against training with the standard CIFAR-10 test set, containing the same images but only with a single label. To perform probabilistic inference, we used SGLD with code adapted from (Zhang et al., 2019). In particular, we used their standard settings of a ResNet18, momentum of 0.9, and a cyclic cosine learning rate schedule. Due to the smaller size of our training set, we used longer cycles (150 epochs rather than 50), and more cycles (8 rather than 4). Importantly, we kept all the parameters of the learning algorithm the same for CIFAR-10H and our CIFAR-10 comparison. The only complication was that to keep the same effective learning rate for CIFAR-10H we need to take into account the number of labellers per datapoint (if we have 50 labels for a datapoint, the loss and the gradients of the loss are 50 times larger, resulting in step-sizes that are also 50 times larger). As such, to keep the same step-sizes, we divided the actual learning rate by 50. An alternative way to look at this is that we use the average log-likelihood per labeller (rather than per-image). Importantly, this change in learning rate, leaves the stationary distribution was unchanged, as changing the learning rate is *not* equivalent to tempering. All other parameters were left at the values specified by Zhang et al. (2019).

As expected, when training on the single-label from the original CIFAR-10 testset (blue lines Fig. 5BC), there are very large tempering effects. In contrast, when training on the ~ 50 labels provided CIFAR-10H (orange), the effects of tempering are far smaller.

Note that while tempering-effects are dramatically reduced, they have not been eliminated entirely. This is expected in our setting, both because of the mismatch between this setup and the exact setup (Sec. 4), but also because the inference method, SGLD, becomes more accurate as the minibatch-size increases, but we can never use full-batch in practical settings due to the large size of datasets such as

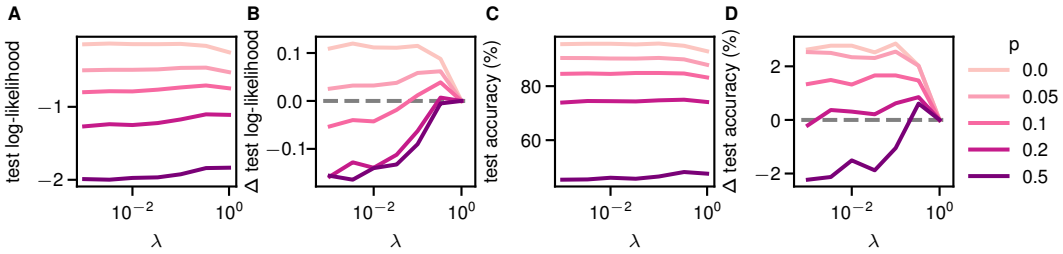

Figure 6: **A** Test log-likelihood with tempering for different probabilities of using a noisy label, p. Noiser labels mean test-log-likelihoods are lower. **B** Change in test log-likelihoods from the baseline at $\lambda = 1$. We see standard cold posterior effects at low noise levels, which reverse at higher noies levels. **C, D** As **A, B** but for test accuracy.

CIFAR-10 (Welling & Teh, 2011). In SGLD minibatch gradients are used as a proxy for the gradient for the full dataset, but these minibatch gradients contain additional noise, and there is a potential that reducing the temperature may partially compensate for this additional noise. This is particularly evident if we consider (Wenzel et al., 2020, Fig. 6), which showed that cold-posterior effects can be amplified by using very small minibatch sizes — smaller minibatches imply larger variance in gradient estimates, and hence more potential for lower temperatures to compensate for that additional noise. That said, these effects do not appear to be significant for CIFAR-10 (Wenzel et al., 2020, Fig. 5), so we are in agreement with Wenzel et al. (2020) that minibatch noise are unlikely to be the primary source of tempering effects.

5.3 CIFAR-10 WITH NOISY LABELS

Curated labels are almost certain to represent the true class, and this is one way to understand the statistical patterns induced by curation that are exploited by tempered posteriors. We therefore considered disrupting this property by adding noise to the labels. In the standard classification setting, this should have very little effect (except to shrink the logit outputs somewhat to give more uncertain outputs). However, and as expected under our theory, we find that adding noise destroys and eventually reverses the cold-posterior effect (Fig. 6). This confirms that cold-posteriors are exploiting specific properties of the labels that are destroyed by adding noise. As such, cold posteriors are not likely to arise e.g. from failing to capture the prior over neural network weights (in which case adding noise to the outputs should have little effect). We suspect that the small improvement in performance at the first value of λ below 1 might be due to partial compensation for additional noise introduced by minibatch estimates of the gradient.

6 RELATED WORK

Wenzel et al. (2020) introduced the cold-posterior effect in the context of neural networks, and proposed and dismissed multiple potential explanations (although none like the one we propose here).

Other past work, though not in the neural network context, argues that tempering may be important in the context of model misspecification (Grünwald, 2012; Grünwald et al., 2017). Critically, we believe that there may be many causes of cold-posterior like effects, including but not limited to curation, model misspecification and artifacts from SGLD. Ultimately, the contribution of each of these factors in any given setting will depend on the exact dataset and inference method in question. Importantly, this also means we do not necessarily expect there to be no tempering in uncurated data, merely that we should see *less* tempering in the case of uncurated data.

Finally, the closest paper is concurrent work (Adlam et al., 2020) raising the possibility that BNNs overestimate aleatoric uncertainty, in part because of high-quality labels available in benchmark datasets. However, they concluded that while cold posteriors might help us to capture our priors, they do not correspond to an exact inference procedure. In contrast, here we give a generative model of dataset curation in which tempered likelihoods emerge naturally even under exact inference methods.

7 CONCLUSIONS

We showed that modelling the process of data-curation can explain the improved performance of tempering or cold posteriors in Bayesian neural networks. That said, we expect that both the misspecification of priors over neural networks weights and inexact inference makes additional contributions to the cold-posterior effect. We hope that our work will prompt more careful dataset and study of data curation. Finally, the same likelihood has much broader applicability, as it also provides an explanation for the effectiveness of common semi-supervised learning methods (Aitchison, 2020).

REFERENCES

Ben Adlam, Jasper Snoek, and Samuel L Smith. Cold posteriors and aleatoric uncertainty. *arXiv preprint arXiv:2008.00029*, 2020.

Laurence Aitchison. A statistical theory of semi-supervised learning. *arXiv preprint*, 2020.

Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. *arXiv preprint arXiv:2002.06470*, 2020.

Juhan Bae, Guodong Zhang, and Roger Grosse. Eigenvalue corrected noisy natural gradient. *arXiv preprint arXiv:1811.12565*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Peter Grünwald. The safe bayesian. In *International Conference on Algorithmic Learning Theory*, pp. 169–183. Springer, 2012.

Peter Grünwald, Thijs Van Ommen, et al. Inconsistency of bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4):1069–1103, 2017.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. *JMLR*, 2015.

Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Toronto, 2009.

Alexander G de G Matthews, James Hensman, Richard Turner, and Zoubin Ghahramani. On sparse variational methods and the kullback-leibler divergence between stochastic processes. *Journal of Machine Learning Research*, 51:231–239, 2016.

Sebastian W Ober and Laurence Aitchison. Global inducing point variational posteriors for bayesian neural networks and deep gaussian processes. *arXiv preprint arXiv:2005.08140*, 2020.

Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz E Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota. Practical deep learning with bayesian principles. In *Advances in Neural Information Processing Systems*, pp. 4289–4301, 2019.

Joshua C Peterson, Ruairidh M Battleday, Thomas L Griffiths, and Olga Russakovsky. Human uncertainty makes classification more robust. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9617–9626, 2019.

Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. *arXiv preprint arXiv:1712.02390*, 2017.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. *arXiv preprint arXiv:1902.03932*, 2019.

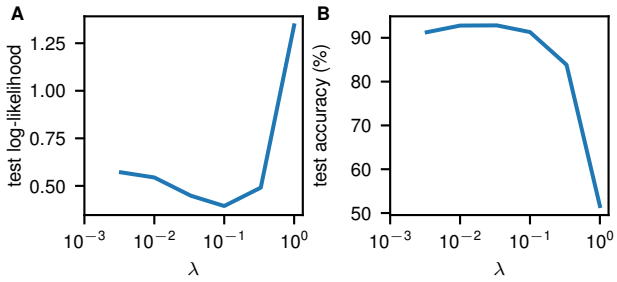

Figure 7: SGD in a ResNet18 with batchnorm removed with a tempered maximum a-posteriori loss. **A** Test log-likelihood. **B** Accuracy.

A COLD POSTERIORS IN MAXIMUM A-POSTERIORI INFERENCE

Our explanation for the cold posterior effect suggests that it is not just due to the stochasticity of the posterior, but should also arise in e.g. maximum a-posteriori (MAP) inference. However, MAP inference causes serious issues in networks with batchnorm. In particular, with batchnorm, the outputs become invariant to the scale of the weights, so the weights can continually decay towards zero (Van Laarhoven, 2017). Note that these considerations are not relevant when we are doing full approximate inference, as the product of the prior and likelihood is a static quantity with a well-defined scale that cannot e.g. decay towards zero. As such, we considered tempering in our usual ResNet18, but where we have deleted the batchnorm layers. We used a standard protocol for these types of network: SGD with momentum of 0.9 and an initial learning rate of 0.1, followed by decays to 0.01 and 0.001 at epochs 150 and 200. We read off performance at the final epoch (250) and used a batch size of 128. We use a standard prior over the weights, with variance $2/\text{fan-in}$, to keep the scale of the inputs and outputs similar.

Remarkably, we found extremely strong cold-posterior effects (Fig. 7). These are perhaps expected as it is common-knowledge (but difficult to find a reference for) that while weight-decay is closely related to MAP inference with a Gaussian prior, good settings for the weight-decay coefficient are much smaller than those suggested by MAP inference with a sensible prior. Interestingly, these results suggest that the robustness of standard models to very low temperatures might arise due to batchnorm, and not be a fundamental property (e.g. having very large amounts of data).

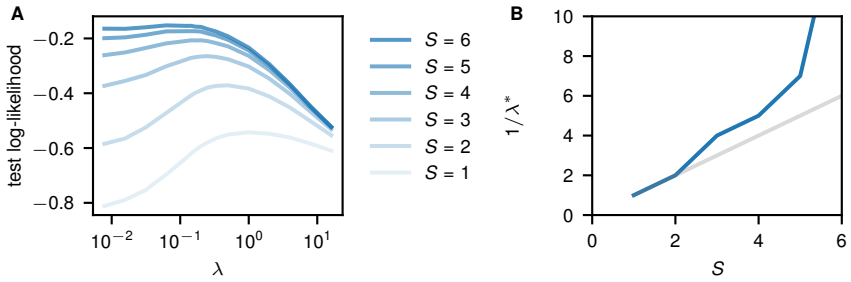

Figure 8: Toy BNN model. **A** Test log-likelihood vs λ for different settings of S. **B** The optimal λ, plotted against S.

B Toy Bayesian neural network example

To confirm the results in the Gaussian processes toy example in the main text, we did the same exercise with a toy Bayesian neural network. In particular, we considered 100 training examples, drawn IID from standard Gaussians, each with 5 input dimensions and 2 class outputs. We generated the "true" or target outputs from a one hidden layer ReLU network, with 30 hidden units, drawn from the prior. We trained the same network, using Langevin dynamics (full batch, but no rejection step; initialized randomly from the prior). As in the GP example, we applied our generative model of dataset curation with different settings for S in the generative model. For test and training, we used the standard single-labeller log-likelihood, which ignores rejected noconsensus inputs. We implemented a highly parallelised Langevin sampling algorithm, which allowed us to draw 500 different datasets, and run one chain for each dataset (and each setting of λ in parallel). The large number of chains allowed us to obtain very accurate results, but did mean that it was not possible to plot lines for individual datasets. We used a learning rate of 0.01 and burn in of 4,000 steps; to compute performance, we averaged over 50 samples of neural-network parameters, which were taken from a chain that was 20,000 steps long in total.

In Fig. 8A, we plot the test-log-likelihood for different settings of λ. There is a clear trend that as S increases, the optimal temperature falls, and at the same time, the curves become flatter. Notably, these curves become flatter at the top for larger settings of S, making it difficult to identify the optimal setting for λ at higher values of S. We then plotted the optimal setting of λ against S (Fig 8B). Initially, $S = 1$ is equivalent to the standard classification setup, as consensus is always reached, and as such, we find that the optimal temperature was 1. As in Fig. 4E, we found that $\lambda^* = 1/S$ held only approximately. Nonetheless, these results suggest that we might need more tempering than suggested by $\lambda^* = 1/S$, which supports our proposed explanation of the cold posterior effect. Finally, it should be noted that these results depend sensitively on the accuracy of the inference algorithm, so it is difficult to say as yet the degree to which mismatch between λ^* and $1/S$ arises due to inaccuracy in inference or would emerge in the intractable true-posterior.