BATCH INVERSE-VARIANCE WEIGHTING: DEEP HETEROSCEDASTIC REGRESSION USING PRIVILEGED INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In model learning, when the training dataset on which the parameters are optimized and the testing dataset on which the model is evaluated are not sampled from identical distributions, we say that the datasets are misaligned. It is well-known that this misalignment can negatively impact model performance. A common source of misalignment is that the inputs are sampled from different distributions. Another source for this misalignment is that the label generating process used to create the training dataset is imperfect. In this work, we consider this setting and additionally assume that the label generating process is able to provide us with a quantity for the role of each label in the misalignment between the datasets, which we consider to be privileged information. Specifically, we consider the task of regression with labels corrupted by heteroscedastic noise and we assume that we have access to an estimate of the variance over each sample. We propose a general approach to include this privileged information in the loss function together with dataset statistics inferred from the mini-batch to mitigate the impact of the dataset misalignment. Subsequently, we propose a specific algorithm for the heteroscedastic regression case, called Batch Inverse-Variance weighting, which adapts Fisher information weighted averaging to the case of neural network function approximation. We demonstrate that this approach achieves a significant improvement in network training performances compared to baselines when confronted with high, input-independent noise.

1 INTRODUCTION

In supervised learning, a central assumption is that the samples in the training dataset, used to train the model, and the samples in the testing set, used to evaluate the model, are sampled from identical distributions. Formally, for input $\mathbf{x}$ and label y, this assumption implies that $p_{\text{train}}(\mathbf{x}, y) = p_{\text{test}}(\mathbf{x}, y)$. This assumption can be decomposed as the product $p_{\text{train}}(\mathbf{x}) \cdot p_{\text{train}}(y|\mathbf{x}) = p_{\text{test}}(\mathbf{x}) \cdot p_{\text{test}}(y|\mathbf{x})$, which is true if two conditions are respected:

1. The features in both datasets are sampled from the same distribution: $p_{\text{train}}(\mathbf{x}) = p_{\text{test}}(\mathbf{x})$. When this is condition is violated, the training dataset is *not representative*.
2. The labels in both datasets are sampled from the same conditional distribution: $p_{\text{train}}(y|\mathbf{x}) = p_{\text{test}}(y|\mathbf{x})$. If this condition is violated, the training labels are *noisy*.

In practice, these assumptions are not always respected because gathering representative and precise data (including labels) can be arduous. In this case, the training and testing datasets are *misaligned*, and the performance of the deployed model may decrease since the training process did not actually optimize the model's parameters based on the correct data (Arpit et al., 2017; Kawaguchi et al., 2020). One possible reason for misalignment is that there is some uncertainty about the labels in the training set as a result of the labeling process. Since our objective is to optimize the performance of the model compared to ground truth labels, we should consider that the labels in test dataset have no uncertainty, even though it may be impossible to collect such a dataset in practice. As a result, $p_{\text{test}}(y|\mathbf{x})$ is sampled from a Dirac delta function, whereas $p_{\text{train}}(y|\mathbf{x})$ is not since it encapsulates the uncertainty in the labelling process, which leads to misalignment.

In this paper, we propose an algorithm for more efficient model training in the case where we have some privileged information about the sample-wise misalignment. More specifically, we examine the case of regression with a deep network where labels are corrupted by heteroscedastic noise. We assume that we have access at least an estimate of the variance of the distribution of the noise that corrupted each label, information that is available if the labels are being generated by some stochastic process that is capable of also jointly reporting uncertainty. We examine how the knowledge of the estimate of the label noise variance can be used to mitigate the effect of the noise on the learning process of a deep neural network. We refer to our method as Batch Inverse-Variance (BIV), which, inspired by information theory, performs a re-weighting using both the the sample-wise variance but also statistics over the entire mini-batch. BIV shows a strong empirical advantage over L2 loss as well as over a simple filtering of the samples based on a threshold over the variance.[1]

Our claimed contributions are threefold:

1. A definition of the problem of learning with privileged information quantifying the misalignment between datasets for the case of heteroscedastic noisy labels in regression.
2. A general formulation of how to use the mini-batch to infer statistics of the dataset and incorporate the privileged information in the loss function when training on neural networks.
3. We present Batch Inverse-Variance as an instantiation of this framework and show its usefulness when applied to regression tasks with labels corrupted by heteroscedastic noise.

The outline of the paper is as follows: In section 2, we describe how the generation of heteroscedastic noisy labels in regression can lead to learning with privileged information, and we explain how the intuition behind Fisher-information weighting can be relevant to this case. In section 3, we position our work among the existing literature on learning with noisy labels. In section 4, we present a general framework to incorporate privileged information regarding dataset misalignment in the mini-batch loss. We introduce BIV within this framework to tackle heteroscedastic regression. In section 5, we describe the setup for the experiments we made to validate the benefits of using BIV, and we present and analyze the results in section 6.

2 BACKGROUND

2.1 HETEROSCEDASTIC NOISY LABELS IN REGRESSION

Here, we introduce how heteroscedastic noisy labels can be generated in regression and how the variance can be known. Consider an unlabelled dataset of inputs $\{\mathbf{x}_i\}$. To label it, one must apply to each input $\mathbf{x}_i$ an instance of a label generator which should provide its true label y_i. This label generator has access to some features $\mathbf{z}_i$ correlated to $\mathbf{x}_i$. We define $LG_j : \mathcal{Z} \rightarrow \mathbb{R}$. When the labelling process is not exact and causes some noise on the label, the noisy label of $\mathbf{x}_i$ provided by LG_j is defined as $\tilde{y}_{i,j}$. Assuming that the noise is unbiased and Gaussian, we can model $\tilde{y}_{i,j}$ as:

$$\tilde{y}_{i,j} = y_i + \delta_{y_{i,j}} \text{ with } \delta_{y_{i,j}} \sim N(0, \sigma^2_{i,j}) \tag{1}$$

$\sigma^2_{i,j}$ can be a function of $\mathbf{z}_i$ and LG_j, without any assumption on its dependence on one or the other. We finally assume that the label generator is able to provide an estimate of $\sigma^2_{i,j}$, therefore being redefined as $LG_j : \mathcal{Z} \rightarrow \mathbb{R} \times \mathbb{R}_{\geq 0}$. The training dataset is formed of triplets $(\mathbf{x}_i, \sigma^2_{i,j}, \tilde{y}_{i,j})$, renamed $(\mathbf{x}_k, \sigma^2_k, \tilde{y}_k)$ for triplet k for simplicity. This setup describes many labelling processes, such as:

Crowd-sourced labelling: In the example case of age estimation from facial pictures, labellers Alice and Bob are given $\mathbf{z}_i = \mathbf{x}_i$ the picture of someone's face and are asked to estimate the age of that person. Age is harder to estimate for older people come (5 and 15 years of age are harder to confuse than 75 and 85) suggesting a correlation between $\sigma^2_{i,j}$ and $\mathbf{z}_i$. But Alice and Bob may also have been given different instructions regarding the precision needed, inducing a correlation between $\sigma^2_{i,j}$ and LG_j. Finally, there may be some additional interactions between $\mathbf{z}_i$ and LG_j, as for example Alice may know Charlie, recognize him on the picture and label his age with lower uncertainty. Both labellers can provide an estimation of the uncertainty around their labels, for example with a plus-minus range which can be used as a proxy for standard deviation.

[1] Our code is available in supplemental material and will be publicly released after the reviewing process.

Labelling from sensor readings or simulations: Imagine you want build a dataset of pictures $\mathbf{x}_i$ from a camera on the ground labelled with the position y_i of a drone in the sky. To estimate the position of the drone at the moment the picture was taken, you could use state estimation algorithms based on the Bayes' filter (Thrun et al., 2006). These algorithms take as an input $\mathbf{z}_i$ the measurements of the drone's sensors, and provide a full posterior distribution over the state, sometimes under a Gaussian assumption for Kalman filters for example. The uncertainty depends, among others, on the precision of the sensors, the observability of a given state, the precision of the dynamic model, and the time since sensor signals were received. Similarly, it is possible to train on simulators, as in climate sciences (Rasp et al., 2018) or in epidemiology (Alsdurf et al., 2020). Some simulators provide with their estimations' uncertainty measures based on the simulation procedure and the inclusion of real measurements in the model.

Using predictions from a neural network in complex neural architectures: In deep reinforcement learning for example, the critic network learns to predict a value from a state-action pair under the supervision of the heteroscedastic noisy output of a target network plus the reward (Mnih et al., 2015; Haarnoja et al., 2018). While the estimation of the uncertainty of the output of a neural network is not an easy task, it is an active field of research (Gal & Ghahramani, 2016; Peretroukhin et al., 2019). There, $\mathbf{z}_i$ is the state-action pair at the next step, and LG_j the target network being updated over time. The prediction is a mix of aleatoric and epistemic uncertainties as defined by Kendall & Gal (2017) which are dependent on both $\mathbf{z}_i$ and LG_j.

2.2 Learning using privileged information

Training with a dataset of triplets $(\mathbf{x}_i, \mathbf{x}_i^*, y_i)$, where $\mathbf{x}_i^*$ is only given at training time and not available at test time, fits in the framework of learning using privileged information (LUPI), defined in Vapnik & Vashist (2009) and mainly applied to SVMs. In most works in this field, this privileged information makes the task easier on a sample-to-sample basis. For example, object detection can be improved by adding segmentation masks (Feyereisl et al., 2014) or depth images (Hoffman et al., 2016). Another interpretation of LUPI is to use privileged information as a vector for knowledge transfer between a teacher and a student (Vapnik & Izmailov, 2015). Hernández-Lobato et al. (2014) and Lambert et al. (2018) have made a link between privileged information and uncertainty, using it to evaluate the confidence of the model for a training sample. The former applied the approach to Gaussian processes through latent noise, and the latter to neural networks through Gaussian dropout.

More formally, at training time the neural network has access to triplets $(\mathbf{x}_k, \mathbf{x}_k^*, y_k)$ where $\mathbf{x}_k$ is the input, y_k its corresponding label, and $\mathbf{x}_k^*$ the additional information with respect to this sample.

The objective in LUPI is the same as classical supervised learning: train the network parameters θ so that, at test time, and without access to information $\mathbf{x}_i^*$, the expected loss is minimized, i.e.:

$$\theta^{opt} = \arg\min_{\theta} \mathbb{E}_{\{\mathbf{x}_i, y_i\} \in D_{\text{test}}} \left[\mathcal{L}\left(f(\mathbf{x}_i, \theta), y_i \right) \right] \tag{2}$$

where $\mathcal{L}(f(\mathbf{x}_i, \theta), y_i)$ is the objective loss function based on the true label and on the network's prediction $f(\mathbf{x}_i, \theta)$, for example the L2 distance in the task of regression.

In our work, we have $\mathbf{x}_i^* = \sigma_i^2$. In contrast with the usual LUPI setting, $\mathbf{x}_i^*$ does not help the task on a sample-to-sample basis, but instead informs about the role of each sample on the misalignment between the datasets due to the noise in the labelling process.

2.3 An information point of view

In information theory, the best estimator $\hat{A}$ for a given quantity A based on noisy measurements a_i is given by an average where each a_i is weighted by its Fisher information, i.e. the information that it carries about A. In the case of Gaussian noise where $a_i \sim \mathcal{N}(A, \sigma_i^2)$, the Fisher information of the measurement is equal to $1/\sigma_i^2$ and the best estimator is:

$$\hat{A} = \left(\sum_i \frac{1}{\sigma_i^2} \right)^{-1} \sum_i \frac{1}{\sigma_i^2} a_i \tag{3}$$

The normalization element $\sum_i 1/\sigma_i^2$ is the information received by the estimator, and is a statistic of the set of measurements. This estimator reaches the Cramer-Rao bound, i.e. the lowest possible bound on its variance given the measurements (Cover & Thomas, 2012).

The case for function approximation in deep neural networks is different. A sample with a ground-truth label where $\sigma_i \approx 0$ does not provide support over the entire test distribution. It should thus not override all the inputs, as it would do in (3). However, the same principle can be applied: *weight stronger the samples that we know are trustworthy, compared to the ones that we know are uncertain.* This is the intuition for Batch Inverse-Variance (BIV) weighting presented in section 4.2.

3 RELATED WORK

Noise on labels amounts to a loss of information. When the noise is significant enough, it leads to overfitting and lower model performance (Liu & Castagna, 1999; Zhang et al., 2017). This effect is more prevalent in small data settings (Van Horn et al., 2015). Four possible strategies exist in the literature to tackle this problem: detection, correction, robustness, or re-weighting. **Detection** consists of identifying noisy labels and ignoring them in the learning process. These methods are often based on the observation that neural networks first fit on consistent, non-noisy data (Arpit et al., 2017), thus converging to a higher loss on the noisy samples (Reed et al., 2015; Shen & Sanghavi, 2019). Other methods use several neural networks to co-teach each other (Han et al., 2018; Yu et al., 2019) or dropout to estimate the consistency of the data (Reed et al., 2015). However, in the case of imbalanced training datasets, higher loss can also be the signature of a non-noisy but rare sample. Cao et al. (2020) address this ambiguity by regularizing different regions of the input space differently. **Correction** strategies go further: once noise is detected, the noisy labels are changed to probability distributions. Such an operation requires a noise model. Goldberger & Ben-Reuven (2017); Kremer et al. (2018); Ma et al. (2018); Tanno et al. (2019); Yi & Wu (2019) learn it jointly with the parameters, assuming a correlation between the noise and the input, the labels, or both. **Robust loss functions** are less sensitive to noise. Liu & Castagna (1999) proposed to avoid overfitting due to noise by ignoring samples during the training when the prediction error is reasonable. More recent works are based on reverse cross-entropy (Wang et al., 2019) or curriculum loss (Lyu & Tsang, 2020). Others leverage a distillate of the information gathered from a subset of clean labels to guide training with noisy labels (Li et al., 2017). **Re-weighting** the samples is another efficient method for mitigating noise in datasets. Liu & Tao (2016) estimate the effective label probabilities as well as noise rates for a given input and use these estimates to weigh the samples using importance sampling. Shu et al. (2019) go one step further by learning the weighting function through a meta-learning method. Jenni & Favaro (2018) control overfitting by adjusting sample weights in the training and validation mini-batches, increasing robustness to overfitting on noisy labels.

While most works that address noisy labels consider classification tasks (Song et al., 2020), only some of these strategies can be generalized to regression. Heteroscedastic regression occurs when each label's noise is sampled from a different distribution. Nix & Weigend (1994) tackle this problem in neural networks by jointly training a variance estimator based on the maximum likelihood of an underlying Gaussian model. Kendall & Gal (2017) use the same idea to estimate the aleatoric (input-dependant) uncertainty of the network's prediction, while using dropout as a Bayesian approximation for the epistemic uncertainty (due to the learning process) as in (Gal & Ghahramani, 2016).

Our method tackles heteroscedastic regression in neural networks using a re-weighting approach. The main distinction between our work and most of the related literature is that, while we do not require that the noise variance is a function of the input or of the label, we do assume that we have access to the noise variance, or at least an estimate of it. In addition, we do not seek to regress the variance of the model's prediction. This is significant compared to the previous works in both regression and classification as it changes the loss function and removes the need for a regularizer.

4 INCORPORATING PRIVILEGED INFORMATION IN THE LOSS FUNCTION

In this section, we first present a general operator to incorporate privileged information and infer dataset statistics on the loss computed at the mini-batch level. Then, we describe our solution, BIV, an instance of this operator with a weighted loss function. Finally, we introduce a more basic filtering function which we will use as a baseline in our experiments.

4.1 General operator for training on the mini-batch level

The value of privileged information about misalignment between the training and testing datasets is often higher when combined with statistics about the datasets. For example, in the case of noisy labels, the uncertainty on each label is relevant when compared to the information carried by the other samples, similarly to (3).

We propose to both incorporate the privileged information and infer dataset statistics during the training of neural networks over a mini-batch, as opposed to the individual samples. There are two main advantages in doing so. First, if the mini-batch samples are independently and identically sampled from the dataset, they can be used to infer some statistics of the whole dataset. This allows us to use such an approach without any pre-processing step over the whole dataset, which is important for many tasks such as continuous learning. Second, by focusing on the single step of computing the loss, this approach does not interfere with any of the variety of other methods used to optimize the learning process, such as regularization, batch normalization, annealing learning rates, etc.

In general, this approach can be expressed by defining an operator $\mathcal{G}$ applied on the objective loss function. For a mini-batch D_i of K sample triplets, we define the loss as:

$$\mathcal{L}_{\text{batch}}(D_i, \theta) = \mathcal{G}\left(\mathbf{x}_{1:K}, \mathbf{x}^*_{1:K}, \mathbf{y}_{1:K}, \mathcal{L}\left(\cdot, \cdot\right)\right) \tag{4}$$

Note that without any privileged information, operator $\mathcal{G}$ is usually the unweighted average of the loss computed over each sample of the batch, which is equivalent to empirical risk minimization.

4.2 Batch Inverse-Variance weighting for heteroscedastic noisy labels

To tackle the problem of heteroscedastic regression with privileged information, we follow the intuition of equation (3) and describe $\mathcal{G}$ as a weighted average with weights $w_k = 1/\left(\sigma_k^2 + \epsilon\right)$. We introduce the Batch Inverse-Variance (BIV) loss function:

$$\mathcal{L}_{\text{batch}}(D_i, \theta) = \left(\sum_{k=0}^{K} \frac{1}{\sigma_k^2 + \epsilon}\right)^{-1} \sum_{k=0}^{K} \frac{\mathcal{L}\left(f(\mathbf{x}_k, \theta), \tilde{y}_k\right)}{\sigma_k^2 + \epsilon} \tag{5}$$

Here, the inverse of the sum has two major roles. It is a normalization constant for the mini-batch, allowing to keep a consistency in the learning rate. It also approximates the information carried by all the samples in the mini-batch, as in (3). Note that consistency is verified as, when σ_k^2 is identical for each sample, this formulation leads to empirical risk minimization.

The hyper-parameter ϵ is effectively a lower bound on the variance. This allows us to incorporate samples with ground-truth labels without completely ignoring the other samples. The choice of ϵ is a trade-off between regulating the weights of near-ground-truth labels and using BIV at its full capacity. We found that it can be set between 0.01 and 0.1 and use $\epsilon = 0.1$. More details on ϵ can be found in appendix B.1.

4.3 Cutoff: filtering heteroscedastic noisy labels

As we do not assume that there is a correlation between $\mathbf{x}_k$ and σ_k^2, most correction and re-weighting algorithms as presented in section 3 are not applicable. Most robust loss function are specifically designed for classification problems. We thus compare BIV to a detection and rejection strategy.

In heteroscedastic regression, an important difference from classification with noisy labels is that all labels are corrupted, albeit not at the same scale. Defining which labels to ignore is therefore a matter of putting a threshold on the variance. As we have access to this information, strategies such as the ones used in section 3 are not necessary. Instead, we simply use an inverse Heaviside step function as a weight in the loss function:

$$w_k = \mathbf{1}_{\sigma_k^2 < C} \tag{6}$$

where the threshold C is a hyper-parameter. Similarly to equation (5), we normalize the loss in the mini-batch by the sum of the weights, equal here to the number of samples considered as valid. As this filtering is equivalent to cutting off a part of the dataset, we refer to this method as 'Cutoff', and consider it to be a relevant baseline to compare BIV against.

5 EXPERIMENTAL SETUP

To test the validity of the BIV loss (5) approach, we compared its performance with the classical L2 loss as well as cutoff loss (6) on two datasets. We refer to ground-truth (GT) labels when training with L2 on noise-less data as the best performance that could be achieved on this dataset.

Unfortunately, we did not find any existing dataset for regression where label uncertainty is associated to the samples. We therefore used two UCI datasets (Dua & Graff, 2017) for regression cases, and artificially added noise to them. UTKFace Aligned&Cropped (Song & Zhang, 2017) (UTKF), is a dataset for image-based age prediction. In the Bike Sharing dataset (Fanaee-T & Gama, 2013), the task is to predict the number of bicycles rented in Washington D.C., from structured data containing the date, hour, and weather conditions. More details about the datasets can be found in appendix A.

5.1 NOISE GENERATION

To produce the datasets $\{\mathbf{x}_k, \sigma_k^2, \tilde{y}_k\}$ with noise as described in section 2.1, we use a two-step process which does not assume any correlation between the noise and the state.

1. the noise variance σ_k^2 is sampled from a distribution $P(\sigma^2)$ which only has support for $\sigma^2 \geq 0$
2. $\tilde{y}_k$ is sampled from a normal distribution $\mathcal{N}(y_k, \sigma_k^2)$.

$P(\sigma^2)$ has a strong effect on the impact of BIV or Cutoff. For example, if it is a Dirac delta and all variances are the same, BIV becomes L2. We evaluate BIV on three different types of $P(\sigma^2)$. The average noise variance μ_P was chosen empirically so that the lowest test loss achieved by L2 is doubled compared to the ground-truth label case: $\mu_P = 2000$ for UTKF and 20000 for BikeSharing.

Uniform distribution The uniform distribution is characterized by its bounds a, b. Its expected value μ_P is the average of its bounds, and its variance $V = (b-a)^2/12$. As P only has support for $\sigma^2 \geq 0$, the maximum variance $V_{\max}$ is when $a = 0$ and $b = 2\mu_P$. While such a distribution is not realistic, it is simple conceptually and allows for interesting insights.

"Binary uniform" A more realistic distribution, which can also help us understand the effects of BIV, is when the data is generated from two regimes: low and high noise. We call the "binary uniform" distribution a mixture of two uniform distributions balanced by parameter p.

With probability p, the label is in a low noise regime: $\sigma^2 \sim U(0, 1)$, with expected value $\mu_l = 0.5$. With probability $1-p$, the label is in a high noise regime: $\sigma^2 \sim U(a_h, b_h)$. a_h and b_h are chosen such that the average is μ_h and the variance of the high-noise distribution is determined by $V_h \in [0, V_{\max}]$. Note that, if we want a given expected value of the whole distribution μ_P, the value of μ_h changes depending on p: $\mu_h = (\mu_P - p\mu_l)/(1-p)$

Therefore, the high-noise expected value μ_h of the noise variance σ^2 in a distribution with high p will be higher than the one for a low p, for the same value of μ_P. In other words, a higher p means more chance to be in the low-noise regime, but the high-noise regime is noisier.

Gamma distributions While the mixture of 2 uniform distributions ensures support in the low noise region, it is not continuous. We therefore also propose to use a Gamma distribution with shape parameter α. If we want to control the expected value μ_P, we adjust $\beta = \alpha/\mu_P$.

For a fixed expected value μ_P, lower α and β mean that there is a stronger support towards low variance noise, but the tail of the distribution spreads longer on the high-noise size. In other words, a lower α means more chances to have low-noise samples, but when they are noisy, the variance is higher. When $\alpha \leq 1$, the highest support is at $\sigma^2 = 0$.

6 EXPERIMENTAL RESULTS AND ANALYSIS

6.1 FOR L2 LOSS, MEAN VARIANCE IS ALL THAT MATTERS

Before looking at the results for BIV, we share an interesting insight for L2 loss with noisy labels which helps simplifying the analysis of the results. Under the unbiased, heteroscedastic Gaussian-based noise model presented in section 5.1, the only parameter of distribution $P(\sigma)$ that mattered to

describe the performance of the L2 loss is its average μ_P, which is also the variance of the overall noise distribution. Independently of the distribution type, and the values of V, p and V_h, or α, as long as μ_P is equal, the L2 loss trained neural networks had the same performance. This is shown in Figure 1. For the sake of clarity, all the curves in this section were smoothed using moving average with a 35 steps window, and the shaded area represents the standard deviation over 3 runs.

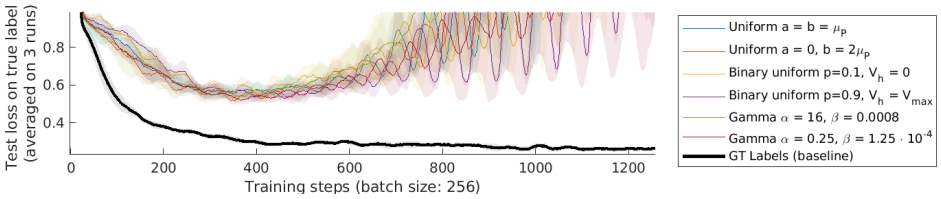

Figure 1: Performance of the neural network on UTKF trained with L2 loss, for different $P(\sigma^2)$ with constant $\mu_P = 2000$. No matter the distribution type or parameters, the performance is similar.

6.2 High and low variance noise regimes: BIV acts as a filter

With the binary uniform distribution, the noise is split in two regimes, with high or low variances. In this case, our results show that BIV performs better than L2, and actually similarly to the cutoff loss presented in section 4.3 with a threshold $C = 1$.

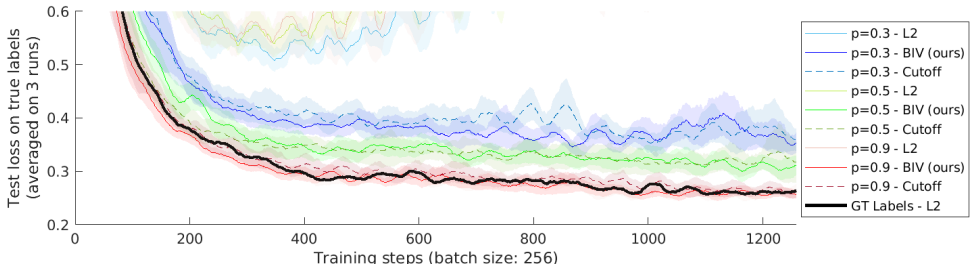

Figure 2: Comparison between BIV, Cutoff and L2 losses for binary uniform distributions of variance with different ps for $\mu_P = 2000$ and $V_h = 0$ on UTKF.

Figure 2 compares the test losses on UTKF with different values of p for $V_h = 0$. While the L2 curves are strongly impacted by the noise, both the BIV and cutoff losses lead to better and very similar performances for a given p. When $p = 0.3$, there are not a lot of information that can be used, and the performance is still impacted by the noise. When $p = 0.9$, there is nearly as much near-ground-truth data as in the noiseless case, and the performance is comparable.

In the case of binary uniform distributions, BIV is acting as a filter, cutting off labels which are too noisy to contain any useful information.

6.3 Continuous, decreasing noise variance distribution: the advantage of BIV

On Gamma distributions, there is no clear threshold to define which information to use. When $\alpha \leq 1$, BIV shows a strong advantage compared to both L2 and cutoff. Figure 3 shows the results in both the BikeSharing and the UTKF datasets for Gamma distributions with $\alpha = 1$.

In both datasets, when the cutoff parameter C is too low ($\mu_P/4$ and $\mu_P/2$), there is not enough data to train the model. When C is too high ($2\mu_P$ and $5\mu_P$), the data is too noisy and the curves go close to the original L2 loss. Even at the best case ($C = \mu_P$), cutoff is not better than BIV. This is because, in contrast to cutoff, BIV is able to extract some information from noisier samples while avoiding to overfit on them.

In Table 1, we present the lowest value of the test loss curves for the different methods with other α parameters for the Gamma distributions over both datasets. BIV consistently leads to the best

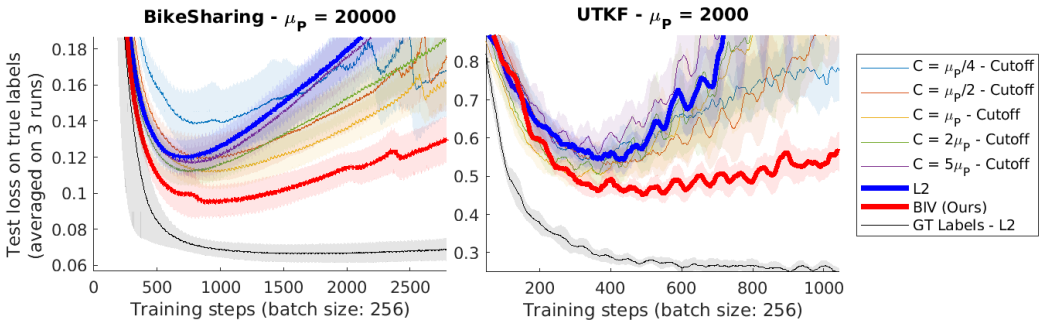

Figure 3: Comparison between the performances of BIV, L2, and different cutoff values on both datasets where the noise variance follows a Gamma distribution with $\alpha = 1$.

performances. It also avoids the need to choose the right cutoff parameter C, which would otherwise need to be optimized. The plots showing these runs can be found in the appendix B.3.1.

Table 1: Lowest test loss for different α on two datasets, for BIV, L2 and several cutoff losses. The test loss with standard deviation is computed as the average over 3 runs. In every case, BIV loss led to the lowest value. The best C value differs based on α.

	$\alpha = 1$		$\alpha = 0.5$		$\alpha = 0.25$	
	UTKF	Bike	UTKF	Bike	UTKF	Bike
$C = \mu_P/20$	0.80±.05	0.356±.048	0.48±.03	0.119±.006	0.37 ±.02	0.088±.004
$C = \mu_P/4$	0.54±.03	0.138±.018	0.44±.02	0.096±.004	0.36±.02	0.083±.005
$C = \mu_P$	0.50±.03	0.112±.010	0.47±.03	0.092±.006	0.43±.03	0.087±.006
$C = 5\mu_P$	0.56±.05	0.116±.009	0.53±.05	0.106±.014	0.51±.02	0.103±.011
L2	0.54±.03	0.119±.010	0.54±.03	0.110±.012	0.52±.02	0.116 ±.009
BIV (ours)	**0.45**±.02	**0.095**±.009	**0.40**±.01	**0.082**±.006	**0.33**±.01	**0.078**±.007
GT labels	0.24±.01	0.066±.005	0.24±.01	0.066±.005	0.24±.01	0.066±.005

The benefit of BIV over L2 is clearly higher when α is lower. This is due to an increase in the support for low-variance noise in $P(\sigma^2)$. The more BIV can count on low-noise elements and differentiate them from high noise ones, the better it can perform. This is consistent with results from section 6.2, and with other experiments we have run. For example, when $\alpha > 1$, the highest support of P is not at $\sigma^2 = 0$. BIV was less able to improve the performance compared to L2.

We also ran the experiment with uniform distributions: the performance is better when variance V is closer to V_{max} (and a to 0). But even when $V = V_{max}$, as there is less support in low noise variance than for Gamma distributions with $\alpha \leq 1$, the improvement is less important.

In all cases, BIV was performing consistently better than L2 and at least better than cutoff in all the experiments we ran. More details on these results can be found in appendix B.2.

7 Conclusion

We have proposed a mini-batch based approach to incorporate in the loss function privileged information which quantifies the participation of each sample to the misalignment between the training and testing datasets. We described how such a setup can occur in the case of regression with heteroscedastic noisy labels. To tackle this problem, we introduced BIV, an information-theory inspired re-weighting scheme. BIV is able to extract more information from the noisy dataset than L2 loss or threshold-based filtering approaches, and consistently outperforms them on both structured and unstructured datasets. BIV can improve the performance of supervised learning in many heteroscedastic regression scenarios, where the label is generated by a process such as crowd-labelling, sensor-based state estimation, simulation, or complex neural architectures. More generally, our framework could be used to account for other types of privileged information quantifying the datasets misalignment, such as under-represented hidden features or correlation between samples.

REFERENCES

Hannah Alsdurf, Yoshua Bengio, Tristan Deleu, Prateek Gupta, Daphne Ippolito, Richard Janda, Max Jarvie, Tyler Kolody, Sekoul Krastev, Tegan Maharaj, and et al. Covi white paper. *arXiv:2005.08502 [cs]*, May 2020. URL http://arxiv.org/abs/2005.08502. arXiv: 2005.08502.

Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and et al. A closer look at memorization in deep networks. *arXiv:1706.05394 [cs, stat]*, Jul 2017. URL http://arxiv.org/abs/1706.05394. arXiv: 1706.05394.

Kaidi Cao, Yining Chen, Junwei Lu, Nikos Arechiga, Adrien Gaidon, and Tengyu Ma. Heteroskedastic and imbalanced deep learning with adaptive regularization. *arXiv:2006.15766 [cs, stat]*, Jun 2020. URL http://arxiv.org/abs/2006.15766. arXiv: 2006.15766.

Thomas M Cover and Joy A Thomas. *Elements of Information Theory*. Wiley, 2 edition, 2012. ISBN 978-1-118-58577-1.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, pp. 1–15, 2013. ISSN 2192-6352. doi: 10.1007/s13748-013-0040-3.

Jan Feyereisl, S. Kwak, Jeany Son, and Bohyung Han. Object localization based on structural svm using privileged information. *Advances in Neural Information Processing Systems*, 1:208–216, 01 2014.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1050–1059. JMLR.org, 2016.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H12GRgcxg.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv:1801.01290 [cs, stat]*, Aug 2018. URL http://arxiv.org/abs/1801.01290. arXiv: 1801.01290.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels. *arXiv e-prints*, art. arXiv:1804.06872, April 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv:1512.03385 [cs]*, Dec 2015. URL http://arxiv.org/abs/1512.03385. arXiv: 1512.03385.

Daniel Hernández-Lobato, Viktoriia Sharmanska, Kristian Kersting, Christoph H. Lampert, and Novi Quadrianto. Mind the nuisance: Gaussian process classification using privileged noise. *arXiv:1407.0179 [cs, stat]*, Jul 2014. URL http://arxiv.org/abs/1407.0179. arXiv: 1407.0179.

Judy Hoffman, Saurabh Gupta, and Trevor Darrell. Learning with side information through modality hallucination. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 826–834. IEEE, Jun 2016. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.96. URL http://ieeexplore.ieee.org/document/7780465/.

Simon Jenni and Paolo Favaro. Deep bilevel learning. *arXiv:1809.01465 [cs, stat]*, Sep 2018. URL http://arxiv.org/abs/1809.01465. arXiv: 1809.01465.

Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *arXiv:1710.05468 [cs, stat]*, Jul 2020. URL http://arxiv.org/abs/1710.05468. arXiv: 1710.05468.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems 30*, pp. 5574–5584. Curran Associates, Inc., 2017.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980 [cs]*, Jan 2017. URL http://arxiv.org/abs/1412.6980. arXiv: 1412.6980.

Jan Kremer, Fei Sha, and Christian Igel. Robust active label correction. volume 84 of *Proceedings of Machine Learning Research*, pp. 308–316, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR. URL http://proceedings.mlr.press/v84/kremer18a.html.

John Lambert, Ozan Sener, and Silvio Savarese. Deep learning under privileged information using heteroscedastic dropout. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8886–8895. IEEE, Jun 2018. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00926. URL https://ieeexplore.ieee.org/document/8579024/.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. *arXiv:1703.02391 [cs, stat]*, Apr 2017. URL http://arxiv.org/abs/1703.02391. arXiv: 1703.02391.

Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):447–461, Mar 2016. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2015.2456899. arXiv: 1411.7718.

Z. P. Liu and J. P. Castagna. Avoiding overfitting caused by noise using a uniform training mode. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339)*, volume 3, pp. 1788–1793 vol.3, 1999.

Yueming Lyu and Ivor W. Tsang. Curriculum loss: Robust learning and generalization against label corruption. *arXiv:1905.10045 [cs, stat]*, Feb 2020. URL http://arxiv.org/abs/1905.10045. arXiv: 1905.10045.

Xingjun Ma, Yisen Wang, Michael E. Houle, Shuo Zhou, Sarah M. Erfani, Shu-Tao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. *arXiv:1806.02612 [cs, stat]*, Jul 2018. URL http://arxiv.org/abs/1806.02612. arXiv: 1806.02612.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, and et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, Feb 2015. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature14236.

D. A. Nix and A. S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 1, pp. 55–60 vol.1, 1994.

Valentin Peretroukhin, Brandon Wagstaff, and Jonathan Kelly. Deep probabilistic regression of elements of so(3) using quaternion averaging and uncertainty injection. In *CVPR Workshops*, 2019.

Stephan Rasp, Michael S. Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39):9684–9689, Sep 2018. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1810286115.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv:1412.6596 [cs]*, Apr 2015. URL http://arxiv.org/abs/1412.6596. arXiv: 1412.6596.

Yanyao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. *arXiv:1810.11874 [cs, stat]*, Feb 2019. URL http://arxiv.org/abs/1810.11874. arXiv: 1810.11874.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *arXiv:1902.07379 [cs, stat]*, Sep 2019. URL http://arxiv.org/abs/1902.07379. arXiv: 1902.07379.

Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *arXiv:2007.08199 [cs, stat]*, Jul 2020. URL http://arxiv.org/abs/2007.08199. arXiv: 2007.08199.

Yang Song and Zhifei Zhang. *UTKFace, Large Scale Face Dataset*, 2017. URL https://susanqq.github.io/UTKFace/. (Accessed June 11, 2020).

Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C. Alexander, and Nathan Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11236–11245. IEEE, Jun 2019. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.01150. URL https://ieeexplore.ieee.org/document/8953406/.

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. Intelligent robotics and autonomous agents. MIT Press, 2006.

Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 595–604. IEEE, Jun 2015. ISBN 978-1-4673-6964-0. doi: 10.1109/CVPR.2015.7298658. URL http://ieeexplore.ieee.org/document/7298658/.

Vladimir Vapnik and Rauf Izmailov. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research*, 16(61):2023–2049, 2015.

Vladimir Vapnik and Akshay Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(5–6):544–557, Jul 2009. ISSN 08936080. doi: 10.1016/j.neunet.2009.06.042.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. *arXiv:1908.06112 [cs, stat]*, Aug 2019. URL http://arxiv.org/abs/1908.06112. arXiv: 1908.06112.

Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. *arXiv:1903.07788 [cs]*, Mar 2019. URL http://arxiv.org/abs/1903.07788. arXiv: 1903.07788.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor W. Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? *arXiv:1901.04215 [cs, stat]*, May 2019. URL http://arxiv.org/abs/1901.04215. arXiv: 1901.04215.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv:1611.03530 [cs]*, Feb 2017. URL http://arxiv.org/abs/1611.03530. arXiv: 1611.03530.

A Appendix - Datasets and Neural Networks

A.1 UTKFace

A.1.1 Dataset description

The UTKFace Aligned&Cropped dataset (Song & Zhang, 2017) consists of 20,000 pictures of faces labelled with their age, ranging from 0 to 116 years. We use it in a regression setting: the network must predict the age of a person given the photo of their face. Unless described otherwise, 16,000 images were used for training, and 4,000 for testing.

Some images are in black and white and some are in color. The pixel dimension of each image is 200x200.

Both the pixels and the labels were normalized before the training, so that their mean is 0 and standard deviation is 1 over the whole dataset. The noise variances were correspondingly scaled, as well as the cutoff threshold if applicable.

A.1.2 NEURAL NETWORK AND TRAINING HYPER-PARAMETERS

The model that we used was a Resnet-18 (He et al., 2015), not pretrained. It was trained with an Adam optimizer (Kingma & Ba, 2017), a learning rate of 0.001 and a batch size of 256, over 20 epochs.

A.2 BIKE SHARING DATASET

A.2.1 DATASET DESCRIPTION

The Bike Sharing Dataset (Fanaee-T & Gama, 2013) consists of 17,379 samples of structured data. It contains, for nearly each hour of the years 2011 and 2012, the date, season, year, month, hour, day of the week, a boolean for it being a holiday, a boolean for it being a working day, the weather situation on a scale of 4 (1: clear and beautiful, 4: stormy or snowy), the temperature, the feeling temperature, the humidity, and the wind speed, in the city of Washington DC. It also contains the number of casual, registered, and total bike renters for each hour as recorded on the Capital Bikeshare system.

We use it in a regression setting: the network must predict the total number of bike renters given the time and weather information. Unless described otherwise, 7,000 samples were used for training, and 3,379 for testing. We used less samples than available for training because the low-data situation, noise has a stronger effect on the performance. The minimal test loss achieved with 7000 noiseless samples was very close to the one with 14000 samples, hinting that the additional samples did not give a lot of additional information.

We applied some pre-processing on the data to make it easier for the network to learn. First, the date was normalized from a scale between day 1 to day 730 to a scale between 0 and 4π. Then, we provided the network with the cosine and the sine of this number. This allowed to have the same representation for the same days of the year, while having the same distance between any two consecutive days, keeping the cyclic nature of a year. A similar idea was applied to hours, normalized from 0 to 2π instead of 0 to 24, and with the cosine and sine given to the network. The day of the week, being a category, was given as a one-hot vector of dimension 7. We also removed the season and the month as it was redundant information with the date.

Overall, the number of features was 19:

1	Year
2-4	Date (sine and cos)
4-5	Hour (sine and cos)
6 to 12	Days of the week (one-hot vector)
13	Holiday boolean
14	Working day boolean
15	Weather situation
16	Temperature
17	Felt temperature
18	Humidity
19	Wind speed

We observed that the network was learning significantly faster and better provided with this format for the data.

Both the features and the labels were normalized before the training, so that their mean is 0 and standard deviation is 1 over the whole dataset. The noise variances were correspondingly scaled, as well as the cutoff threshold if applicable.

A.2.2 NEURAL NETWORK AND TRAINING HYPER-PARAMETERS

The model that we used was a multi-layer perceptron with 4 hidden layers, the first one with 100 neurons, then 50, 20, and 10. The activation function was ReLU. We did not use any additional technique such as batch normalization as it did not improve the performances.

The model was trained over 100 epochs on mini-batches of size 256, using the Adam optimizer with learning rate 0.001.

B APPENDIX - ADDITIONAL EXPERIMENTS

B.1 THE INFLUENCE OF ϵ

In this section, we provide experimental results justifying our recommendation of the range of $[10^{-2}; 10^{-1}]$ for ϵ. In the experiments presented in this article, we have used $\epsilon = 10^{-1}$. ϵ must be chosen as part of a trade-off between mitigating the effect of BIV with near ground-truth labels while keeping its effect with noisy labels. To better understand these results, it is important to remember that ϵ is added to the variance that is used in the loss function. When the labels are normalized - which is the case in our work -, the noise and its variance for each label is normalized too. The value we recommend for ϵ should therefore be valid for any normalized set of labels.

B.1.1 ϵ FOR BIV WITH NEAR GROUND-TRUTH LABELS

One of the main problems BIV induces is in the presence of ground-truth or near ground-truth (NGT) labels. In this case, a near-zero variance can induce an very strong weight, effectively reduce the mini-batch to this single sample, and thus ignore the other potentially valid samples.

In this case, hyperparameter ϵ is key, as it allows to set a maximal weight and has a stronger relative effect on the NGT labels than on noisier ones. We tested on the UTKF dataset with a uniform distribution of variance from $a = 0$ to $b = 1$, which is effectively very little noise for the L2 loss but, by allowing NGT labels, can already showcase the influence of ϵ on the BIV performance. The resulting graph can be seen in figure 4.

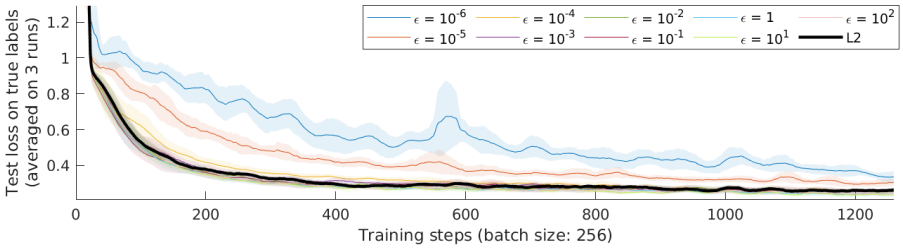

Figure 4: Impact of ϵ when training with NGT labels using BIV loss on UTKF dataset. The variance was sampled through a uniform distribution bound between 0 and 1. Very low ϵ shows a drastic loss of performance compared to L2 loss, as NGT labels monopolize the weights in the mini-batch.

It is clear that a very small ϵ, such as 10^{-6} or 10^{-5}, leads to a significant loss of performance. However, over 10^{-2}, the performance is as good as the L2 loss. Note that it is not surprising when ϵ is high such as 10^2, as in this case the weights are very close to similar and BIV is effectively the same as L2.

B.1.2 A HIGH ϵ REDUCES THE ADVANTAGES OF BIV ON NOISY LABELS

As discussed in the previous section, a higher value of ϵ makes the weights more similar for each of samples, and therefore reduces the effect of BIV. We tested BIV with different values of ϵ with a binary distribution with $p = 0.5$ and $\mu_P = 2000$ on UTKF. In this setup, BIV should have enough data to train correctly when filtering out the labels in the noisy regime, as shown in section 6.2. The results are shown in figure 5.

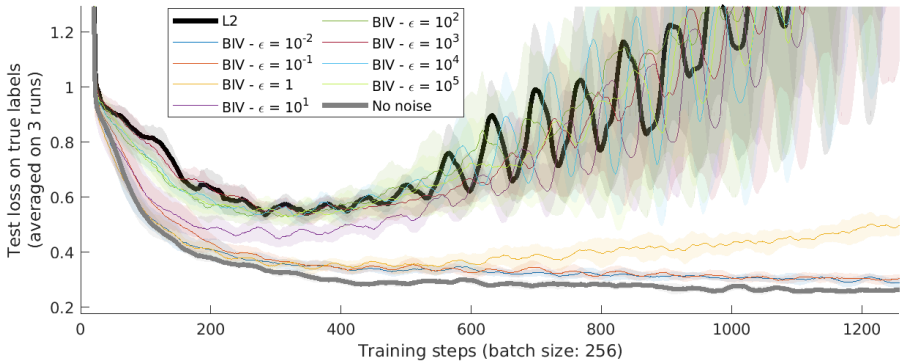

Figure 5: Impact of ϵ when training with NGT labels using BIV loss on UTKF dataset. The variance was sampled through a uniform distribution bound between 0 and 1. Very low ϵ shows a drastic loss of performance compared to L2 loss, as NGT labels monopolize the weights in the mini-batch.

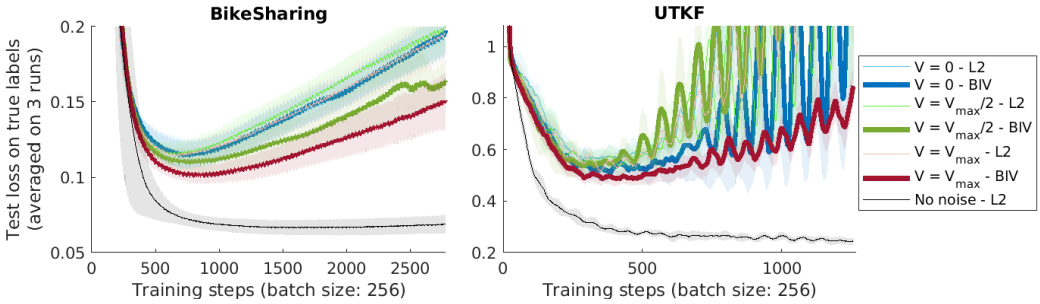

Figure 6: Test loss for L2 and BIV learning on uniform with different variances V.

As expected, when ϵ is very high, the results are very similar to L2. The first effects of BIV can be seen when $\epsilon = 10$, but until $\epsilon = 1$ it is still not optimal. From $\epsilon = 0.1$, the algorithm is correctly filtering the data.

Considering the results from sections B.1.1 and B.1.2, we consider that ϵ should be between 10^{-2} and 10^{-1} to optimize the balance between regulating the importance of near ground-truth labels and benefiting from the effect of BIV.

B.2 BIV ON DIFFERENT DISTRIBUTIONS

B.2.1 UNIFORM DISTRIBUTIONS

We present in figure 6 the results of the experiment with uniform distributions in more details.

As explained in section 6.3, we observe that BIV and L2 have the same performances when $V = 0$ (and $a = b = \mu_P$). This is to be expected, as all samples have the exact same noise variance and thus the same weights. When $V = V_{max}$ ($a = 0$ and $b = 2\mu_P$), BIV has an advantage, as it is able to differentiate the samples and use the support of low-noise labels. When $V = V_{max}/2$ ($a = 0.293\mu_P$ and $b = 1.707\mu_P$), the difference between the samples is less important, and BIV only does a bit better than L2 on BikeSharing. On UTKF, the process has more variability and it is difficult to detect this effect.

In all cases, the benefit from using BIV is less important than with Gamma distributions with $\alpha \leq 1$, where the support on low-noise samples is higher.

In this setting, we also show as shown in figure 7 that cutting off the noisy data is not a good strategy, as it always performs worse than L2.

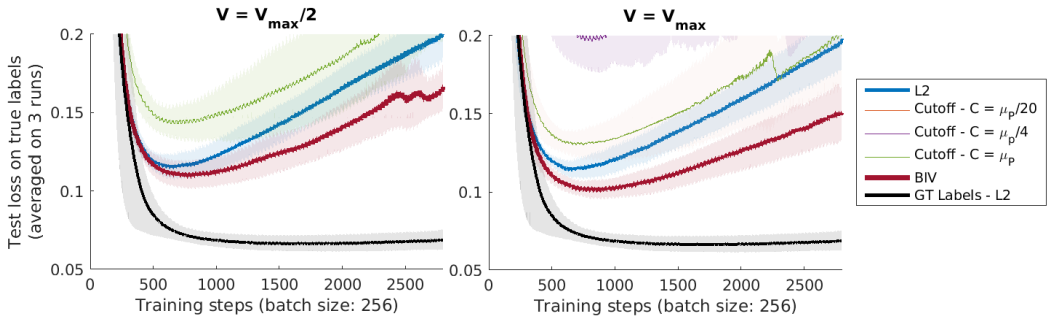

Figure 7: On BikeSharing with $\mu_P = 20000$, using cutoff is not helpful in the uniform setting. This is due to the significant loss of information induced by such a strategy.

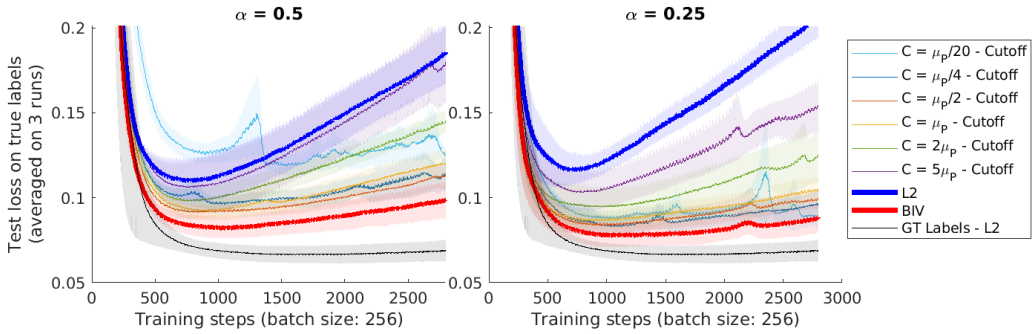

Figure 8: Test loss on the BikeSharing dataset, with $\alpha \leq 1$

B.3 BIV on Gamma distributions

B.3.1 $\alpha \leq 1$

As described in section 6.3, the smaller α, the better the performance of BIV and cutoffs. We show in figures 8 and 9 the curves that led to the numbers in Table 1. BIV consistently outperforms the other methods. The performance of cutoff methods strongly depends on C, and the best value of C is not the same for every distribution P.

B.3.2 $\alpha > 1$

When $\alpha > 1$, the highest support of $P(\sigma^2)$ shifts towards μ_P. This makes the samples less distinguishable for BIV and therefore the benefits of using it are reduced. This is shown in Figure 10 on UTKF.

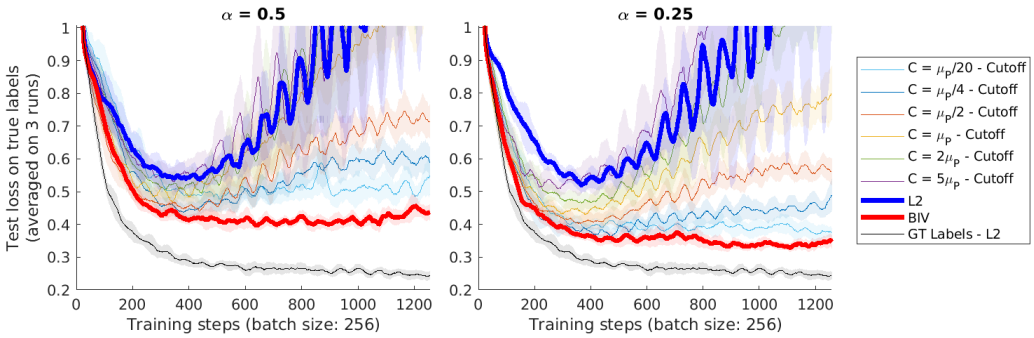

Figure 9: Test loss on the UTKF dataset, with $\alpha \leq 1$

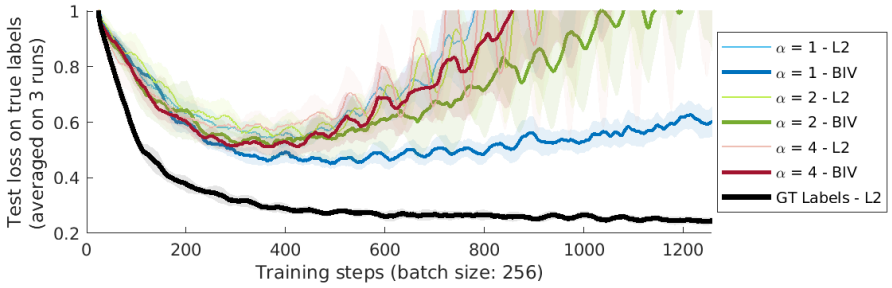

Figure 10: Test loss on the UTKF dataset for L2 and BIV learning on Gamma function with $\alpha \geq 1$.