Repositorio Institucional de la Universidad Autónoma de Madrid
https://repositorio.uam.es

Esta es la **versión de autor** del artículo publicado en:
This is an **author produced version** of a paper published in:

Neurocomputing 471 (2022): 260–274

DOI: https://doi.org/10.1016/j.neucom.2020.09.076

Copyright: © 2020. This manuscript version is made available under the CC-BY-NC-ND 4.0 licence http://creativecommons.org/licenses/by-nc-nd/4.0/

El acceso a la versión del editor puede requerir la suscripción del recurso

Access to the published version may require subscription

Adversarial α-divergence Minimization for Bayesian Approximate Inference

Simón Rodríguez Santana[a,*], Daniel Hernández-Lobato[b]

[a] *Institute of Mathematical Sciences (ICMAT-CSIC), Campus de Cantoblanco, C/Nicolás Cabrera, 13-15, 28049 Madrid, Spain.*
[b] *Escuela Politécnica Superior, Universidad Autónoma de Madrid, Campus de Cantoblanco, C/Francisco Tomás y Valiente 11, 28049 Madrid, Spain.*

Abstract

Neural networks are state-of-the-art models for machine learning problems. They are often trained via back-propagation to find a value of the weights that correctly predicts the observed data. Back-propagation has shown good performance in many applications, however, it cannot easily output an estimate of the uncertainty in the predictions made. Estimating the uncertainty in the predictions is a critical aspect with important applications. One method to obtain this information consists in following a Bayesian approach to obtain a posterior distribution of the model parameters. This posterior distribution summarizes which parameter values are compatible with the observed data. However, the posterior is often intractable and has to be approximated. Several methods have been devised for this task. Here, we propose a general method for approximate Bayesian inference that is based on minimizing α-divergences, and that allows for flexible approximate distributions. We call this method adversarial α-divergence minimization (AADM). We have evaluated AADM in the context of Bayesian neural networks. Extensive experiments show that it may lead to better results in terms of the test log-likelihood, and sometimes in terms of the squared error, in regression problems. In classification problems, however, AADM gives competitive results.

[*] Corresponding author
Email addresses: `simon.rodriguez@icmat.es` (Simón Rodríguez Santana), `daniel.hernandez@uam.es` (Daniel Hernández-Lobato)

Preprint submitted to Neurocomputing *July 14, 2020*

Keywords: Bayesian Neural Networks; Approximate Inference; Alpha Divergences; Adversarial Variational Bayes

1. Introduction

In the past years, Neural Networks (NNs) have become very popular due to the good empirical achievements in a wide variety of learning problems. Specifically, Deep Neural Networks (DNNs) trained with back-propagation have significantly improved the state-of-the-art in supervised learning tasks [1]. Moreover, variations of the simple original NN models have been specifically designed to take advantage of underlying structure on the input data. This is the case for Convolutional Neural Networks (CNNs) [2] or Long-Short Term Memory Networks (LSTMs) [3], both of which represent some of the best performing models for dealing with structured data such as images and texts, respectively. NNs can be trained on Graphical Processing Units (GPUs), which significantly reduces the total training time and the effort needed to produce highly accurate results. These models can therefore be trained on huge amounts of data very quickly, showing excellent results in regression and a competitive performance also in classification tasks. In spite of the advantages described, the good performance results come with some drawbacks, such as the concerns about over-fitting due to the high number of parameters to be adjusted, or the lack of a confidence measure on the predicted outputs associated to the input data [4]. More precisely, regular NNs only produce point-estimate predictions and do not provide any information about the certainty of such outcome. Even in multi-class problems where the results are given in terms of a soft-max function which outputs probabilities, it is important to keep in mind that the output values do not correspond to the confidence of the prediction. In particular, a high class label probability may correspond to a data instance that will be often misclassified by the network.

The problems described can be addressed by following a Bayesian approach in the training process, instead of relying on back-propagation for finding point-estimates of the model parameters. One of the main features of Bayesian probabilistic models such as Bayesian neural networks (BNNs) [5] is that they are able to capture the uncertainty in the model parameters (the network weights) and the effects it produces in the final predictions, therefore providing an estimate of the models' ignorance on the input data in each specific case. This extra output information can be used in different ways: for example, confronting problems in artificial intelligence safety, performing active learning, or dealing with possible adversaries which may manipulate the data [4]. Therefore, uncertainty estimates associated to the model predictions can be very important to make optimal decisions when dealing with input data that the machine learning algorithm has never seen before.

The Bayesian approach relies on computing a posterior distribution for the model parameters given the data [4]. This posterior distribution is obtained using Bayes' rule simply by multiplying a likelihood function (which captures how well specific values of the parameters explain the observed data) and a prior distribution (which includes prior knowledge about what potential values these parameters may take). This posterior distribution summarizes which model parameters (*i.e.*, the neural network weights) are compatible with the observed data. Intuitively, if the model is rather complex, the posterior will be very broad. By contrast, if the model is fairly simple, the posterior will concentrate on a specific region of the parameters space. The information contained in the posterior distribution can be readily translated into a predictive distribution which carries information about the uncertainty on the predictions made. For this, one simply has to average the predictions of the model for each parameter configuration weighted by the corresponding posterior probability.

A difficulty of the Bayesian approach is, however, that computing the posterior distribution is intractable for most problems. Therefore, in practice, one has to resort to approximate methods. Most of these methods approximate the exact posterior using an approximate distribution q. The parameters of q are tuned by minimizing a divergence between q and the exact posterior. This is how methods such as variational inference (VI), expectation propagation (EP) or black-box-α work in practice [6, 7, 8]. Although these methods are very fast and scalable, they often suffer from the lack of flexibility of the approximate distribution q, which is typically set to be a parametric distribution that cannot adequately match the exact posterior. Therefore, these methods may suffer from strong approximation bias. Importantly, a poor approximation of the exact posterior is expected to lead to a worse predictive distribution, less accurate predictions, and a worse estimate of the uncertainty in the predictions made.

Recently, several methods have been proposed to increase the flexibility of the approximate distribution q [9, 10, 11, 12, 13]. Among these, a successful approach is to use an implicit model for the approximate distribution [14]. Under this setting, q is obtained by applying an adjustable non-linear function (*e.g.*, given by the output of a neural network) to a source of Gaussian noise. If the non-linear function is flexible enough, almost any distribution can be approximated like this. Nevertheless, even though q is a distribution that is easy to sample from, its p.d.f. can not be obtained analytically due to the complexity of the non-linear function. More precisely, marginalizing the Gaussian noise is intractable. This makes approximate inference (*i.e.*, tuning the parameters of the non-linear function) very challenging. Adversarial variational Bayes (AVB) is a technique that solves this problem by minimizing the Kullback-Leibler (KL) divergence between q and the exact posterior [10]. This technique avoids evaluating the p.d.f. of q by learning a discriminator network that estimates the log-ratio between the posterior approximation q and the prior distribution over the model parameters.

AVB and also other methods such as VI or EP (only locally and in the reversed way) rely on minimizing the KL divergence between the approximate distribution q and the exact posterior. The α-divergence generalizes the KL divergence and includes a parameter $\alpha \in (0, 1]$ that can be adjusted. In particular, when $\alpha \to 0$, the α-divergence tends to the KL-divergence optimized by VI. By contrast, if $\alpha = 1$, the α-divergence is the reversed KL-divergence, *i.e.*, the KL-divergence between the exact posterior and q, which is locally optimized by EP. Recently, it has been empirically shown that one can obtain better results, in terms of the approximate predictive distribution, by minimizing α-divergences locally using intermediate values of the α parameter in the case of parametric q [7]. However, it is not clear if one can also obtain better results in the case of using implicit models for q, such as the one considered by AVB.

In this paper we extend AVB to locally minimize α-divergences, in an approximate way (instead of the regular KL divergence) with α an adjustable parameter. We refer to such a method as Adversarial α-divergence minimization (AADM). Therefore, AADM can be seen as a generalization of AVB that allows to optimize a more general class of divergences, resulting in flexible approximate distributions q with different properties. When $\alpha \to 0$, AADM targets the same objective as AVB. When $\alpha = 1$, AADM is similar to EP with a flexible approximate distribution q. Intermediate values of α result in different properties of the approximate distribution. We have evaluated AADM in the context of Bayesian Neural Networks and tested different values of the α parameter. The experiments carried out involve several regression and classification problems extracted from the UCI repository, the MNIST dataset and the CIFAR-10 dataset. The experiments show that in regression problems one can obtain, in general, better prediction results than those of AVB and standard VI by using intermediate values of α. In particular, the mean squared error, test log-likelihood, and other performance metrics of the predictive distribution such as the *continuous ranked probability score* (CRPS) [15] improve when intermediate values of α are used. We have also evaluated AADM in the context of binary and multi-class classification problems. In these cases, however, we have observed that AADM gives similar results to those of AVB, both in terms of the prediction error and the test log-likelihood, as well as in terms of other performance metrics based on the *Brier* score [15].

2. Variational Inference and Adversarial Variational Bayes

Adversarial Variational Bayes (AVB) [10] is an extension of variational inference (VI) [16] that allows for implicit models for the approximate distribution q. We will briefly introduce here first VI and then AVB.

2.1. Variational Inference

Let $\mathbf{w}$ be the latent variables of the model, *e.g.*, the neural network weights. The task of interest in VI is to approximate the posterior distribution of $\mathbf{w}$ given the observed data. For simplicity we will focus on regression models, but the method is broadly applicable to any model and is not limited to neural networks.

Consider a training set $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i$ is some d-dimensional input vector and $y_i \in \mathbb{R}$ is the associated target value. The posterior distribution is given by Bayes' rule:

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathbf{y}|\mathbf{w},\mathbf{X})p(\mathbf{w})}{p(\mathcal{D})} = \frac{\left[\prod_{i=1}^N p(y_i|\mathbf{w},\mathbf{x}_i)\right] p(\mathbf{w})}{p(\mathcal{D})}\,, \tag{1}$$

where $\mathbf{X}$ is a matrix with the observed vectors of input attributes and $\mathbf{y} = (y_1, \ldots, y_N)^{\mathrm{T}}$. We have assumed i.i.d. data and hence, the likelihood factorizes as $p(\mathbf{y}|\mathbf{w},\mathbf{X}) = \prod_{i=1}^N p(y_i|\mathbf{w},\mathbf{x}_i)$. In (1) $p(\mathbf{w})$ is the prior distribution of the latent variables of the model (*i.e.*, the neural network weights) and $p(\mathcal{D}) = \int p(\mathbf{y}|\mathbf{w},\mathbf{X})p(\mathbf{w})d\mathbf{w}$ is just a normalization constant. In the case of regression problems $p(y_i|\mathbf{w},\mathbf{x}_i)$ is often a Gaussian distribution, *i.e.*, $\mathcal{N}(y_i|f(\mathbf{x}_i),\sigma^2)$, where $f(\mathbf{x}_i)$ is the output of the neural network and σ^2 is the variance of the output noise. In the case of binary classification problems, $p(y_i|\mathbf{w},\mathbf{x}_i)$ is given by the sigmoid activation function. In multi-class problems, the soft-max function is used instead. Furthermore, $p(\mathbf{w})$ is often a factorizing Gaussian with zero mean and variance σ_0^2 (see *e.g.*, [6, 7, 8]). Given (1) the predictive distribution of the model for the label $y^\star$ of a new test point $\mathbf{x}_\star$ is:

$$p(y^\star|\mathcal{D}) = \int p(y^\star|\mathbf{w},\mathbf{x}^\star)p(\mathbf{w}|\mathcal{D})d\mathbf{w}\,. \tag{2}$$

In the case of regression problems, the model prediction would be the expected value of $y^\star$ under (2) and the confidence in the prediction can be estimated, *e.g.*, by the standard deviation. In classification problems, the model prediction would be a probability for each class label (which takes into account the uncertainty about $\mathbf{w}$). In practice, $p(\mathbf{w}|\mathcal{D})$ is intractable because $p(\mathcal{D})$ has no closed form expression and one has to use an approximation to this distribution in (2).

VI [16] approximates (1) using a parametric distribution $q(\mathbf{w})$ which is often a factorizing Gaussian $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma}$ a diagonal matrix. Let ϕ be the set of parameters of $q(\mathbf{w})$, *i.e.*, $\phi = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$. These parameters are adjusted to minimize the KL divergence between $q(\mathbf{w})$ and the exact posterior (1). Consider the following decomposition of $\log p(\mathcal{D})$:

$$\log p(\mathcal{D}) = \mathbb{E}_{q_\phi(\mathbf{w})}[\log p(\mathbf{y}, \mathbf{w}|\mathbf{X}) - \log q(\mathbf{w})] + \mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D}))\,, \tag{3}$$

where $\mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D}))$ is the KL divergence between $q(\mathbf{w})$ and the exact posterior:

$$\mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D})) = -\int q_\phi(\mathbf{w}) \log \frac{p(\mathbf{w}|\mathcal{D})}{q_\phi(\mathbf{w})} d\mathbf{w} \geq 0\,. \tag{4}$$

The KL divergence is always non-negative and is only zero if the two distributions are the same. Therefore, by minimizing this divergence VI enforces that $q(\mathbf{w})$ looks similar to the exact posterior (1).

Since $\log p(\mathcal{D})$ in (3) is a constant term independent of ϕ, the KL divergence in (4) can be minimized by maximizing the first term in the r.h.s. of (3) with respect to ϕ. This term is often referred to as the evidence lower bound:

$$\begin{aligned}\mathcal{L}(\phi) &= \mathbb{E}_{q_\phi(\mathbf{w})}[\log p(\mathbf{y}, \mathbf{w}|\mathbf{X}) - \log q(\mathbf{w})] \\ &= \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{w})}[p(y_i|\mathbf{w}, \mathbf{x}_i)] - \mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}))\,,\end{aligned} \tag{5}$$

where $\mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}))$ is the KL divergence between $q(\mathbf{w})$ and the prior $p(\mathbf{w})$. In some cases there exists a closed-form expression for this divergence, *e.g.* if $q(\mathbf{w})$ and the prior are Gaussian, which is usually the case for VI [17]. The maximization of (5) can be done using stochastic optimization techniques that sub-sample the training data and that approximate the required expectations using Monte Carlo samples (see [8] for further details). The hyper-parameters of the model, *i.e.*, the noise and prior variance σ^2 and σ_0^2 are estimated by maximizing $\mathcal{L}(\phi)$, which approximates $\log p(\mathcal{D})$ since $\text{KL}(q(\mathbf{w})||p(\mathbf{w}|\mathcal{D}))$ is expected to be fairly small. Finally, after training, the posterior approximation can replace the exact posterior in (2) and the predictive distribution for new data can be approximated by a Monte Carlo average over the posterior samples.

2.2. Adversarial Variational Bayes

AVB extends VI to account for implicit models for the approximate distribution $q(\mathbf{w})$ [10]. An implicit model for $q(\mathbf{w})$ is is a distribution that is easy to generate samples from, but that lacks a closed form expression for the p.d.f. An example is a source of standard Gaussian noise that is non-linearly transformed by a neural network. That is,

$$q_\phi(\mathbf{w}) = \int \delta\left(\mathbf{w} - \mathbf{f}_\phi(\boldsymbol{\epsilon})\right) \mathcal{N}(\boldsymbol{\epsilon}|\mathbf{0}, \mathbf{I}) d\boldsymbol{\epsilon} \,, \tag{6}$$

where $\mathbf{f}_\phi(\boldsymbol{\epsilon})$ is the output of a neural network that receives $\boldsymbol{\epsilon}$ at the input and $\delta(\cdot)$ is a delta function. In general, the integral in (6) is intractable due to the strong non-linearities of the neural network. Nevertheless, it is very easy to generate $\mathbf{w} \sim q_\phi$. For this, one only has to generate $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to then compute $\mathbf{w} = \mathbf{f}_\phi(\boldsymbol{\epsilon})$. If the noise dimension is large enough and $\mathbf{f}_\phi(\cdot)$ is flexible enough, any probability distribution can be described like this. Therefore, an implicit model may reduce the bias in VI associated with the parametric distribution $q(\mathbf{w})$.

Using an implicit distribution in VI is challenging because the lower bound in (5) cannot be easily evaluated nor maximized. The reason for this is that the evaluation of the KL divergence term requires the p.d.f. of $q(\mathbf{w})$. AVB provides an elegant solution to this problem. For this, the KL term is expressed as:

$$\text{KL}(q(\mathbf{w})||p(\mathbf{w})) = \mathbb{E}_{q_\phi(\mathbf{w})}\left[\log q_\phi(\mathbf{w}) - \log p(\mathbf{w})\right] = \mathbb{E}_{q_\phi(\mathbf{w})}\left[T(\mathbf{w})\right] \,, \tag{7}$$

where $T(\mathbf{w})$ is simply the log-ratio between q_ϕ and the prior. AVB estimates $T(\mathbf{w})$ as the output of another neural network that discriminates between samples of $\mathbf{w}$ generated from q_ϕ and from the prior [10]. This technique has also been considered in other works [13, 18, 14]. Let $T_\omega(\cdot)$ be the output of the discriminator. The following objective is considered in AVB for finding the optimal discriminator, assuming $q_\phi(\mathbf{w})$ is fixed:

$$\max_{\omega} \quad \mathbb{E}_{q_\phi(\mathbf{w})}\left[\log \sigma(T_\omega(\mathbf{w})) + \mathbb{E}_{p(\mathbf{w})}[\log(1 - \sigma(T_\omega(\mathbf{w})))]\right] \,, \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid-function. Roughly speaking, this objective tries to make the discriminator differentiate between samples generated from $q_\phi(\mathbf{w})$ and from the prior $p(\mathbf{w})$.

In [10] it is shown that the optimal discriminator $T_{\omega^\star}$ for (8) is precisely

$$T_{\omega^\star}(\mathbf{w}) = \log q_\phi(\mathbf{w}) - \log p(\mathbf{w}), \tag{9}$$

which is the result desired to correctly estimate the KL divergence between q_ϕ and the prior. See the supplementary material for further details. In particular, the discriminator can be plugged in (7) and the expectation can be approximated simply by a Monte Carlo average by generating samples from q_ϕ.

Given $T_{\omega^\star}$, the lower bound employed in AVB is obtained by re-writing the evaluation of the KL divergence between q_ϕ and the prior:

$$\mathcal{L}(\phi) = \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{w})}[p(y_i|\mathbf{w}, \mathbf{x}_i)] - \mathbb{E}_{q_\phi(\mathbf{w})}[T_{\omega^\star}(\mathbf{w})] \,. \tag{10}$$

Note that all the required expectations can be simply approximated by generating samples from q_ϕ and the sum across the training data can be approximated using mini-batches. This lower bound can be hence easily maximized w.r.t. ϕ using stochastic optimization techniques. For this, however, we need to differentiate the stochastic estimate of the objective with respect to ϕ. This may seem complicated since $T_{\omega^\star}(\mathbf{w})$ is defined as the solution of an auxiliary optimization problem that depends on ϕ. However, as shown in [10], due to the expression for the optimal discriminator, $\mathbb{E}_{q_\phi(\mathbf{w})}\left(\nabla_\phi T_{\omega^\star}(\mathbf{w})\right) = 0$. Therefore the dependence of $T_{\omega^\star}(\mathbf{w})$ w.r.t ϕ can be ignored. In practice, both q_ϕ and the discriminator

$T_\omega(\mathbf{w})$ are trained simultaneously. Nonetheless, q_ϕ is updated by maximizing (10) using a smaller learning rate than the one used to update the discriminator T_ω, which considers the objective in (8). This helps to guarantee that T_ω is an accurate estimator of the log-ratio between q_ϕ and the prior, and that the KL divergence is correctly estimated when updating q_ϕ.

Finally, the performance of AVB depends on achieving a good approximation $T_\omega(\mathbf{w})$ to the optimal discriminator. However, in practice the approximation may not be sufficiently close to the optimum. This can be caused by the fact that the approximate posterior distribution and the prior are very different from each other. This may result in a *relaxed* discriminator which can distinguish with ease the samples from both distributions, but fails to estimate accurately the log-ratio between the p.d.f's. To address this issue, in [10] it is proposed a technique called *adaptive contrast*, which consists in introducing a new auxiliary conditional probability distribution $r_\alpha(\mathbf{w})$ with known density that approximates q_ϕ. This auxiliary distribution is set to be a factorizing Gaussian whose mean and variances match those of q_ϕ. When this distribution is introduced in the objective of VI, the term $\text{KL}(q(\mathbf{w})||p(\mathbf{w}))$ is replaced by the term $\text{KL}(q(\mathbf{w})||r_\alpha(\mathbf{w}))$, which can be estimated again using a classifier to discriminate samples from $q(\mathbf{w})$ and samples from $r_\alpha(\mathbf{w})$. This, in general, results in an improvement in the final performance of the algorithm. For further details about adaptive contrast please see the supplementary material.

3. Alpha Divergence Minimization

Before describing the proposed method, we briefly review here the α-divergence, of which we make extensive use. Let p and q be two distributions over the vector $\boldsymbol{\theta}$. The α-divergence between p and q is non-negative and only equal to zero if $p = q$ [19]. The corresponding expression is given by

$$D_\alpha[p|q] = \frac{1}{\alpha(1-\alpha)}\left(1 - \int p(\boldsymbol{\theta})^\alpha q(\boldsymbol{\theta})^{1-\alpha} d\boldsymbol{\theta}\right) . \qquad (11)$$

This divergence has a parameter $\alpha \in \mathbb{R} \setminus \{0, 1\}$. Depending on the value of α it recovers different well-known divergences between probability distributions. For example,

$$D_1[p|q] = \lim_{\alpha \to 1} D_\alpha[p|q] = \mathrm{KL}[p||q]\,, \tag{12}$$

$$D_0[p|q] = \lim_{\alpha \to 0} D_\alpha[p|q] = \mathrm{KL}[q||p]\,, \tag{13}$$

$$D_{\frac{1}{2}}[p|q] = 2\int \left(\sqrt{p(\boldsymbol{\theta})} - \sqrt{q(\boldsymbol{\theta})}\right)^2 d\boldsymbol{\theta} = 4\mathrm{Hel}^2[p|q]\,. \tag{14}$$

The first two limiting cases given by (12) and (13) represent the two different possibilities for the KL-divergence between distributions. Moreover, (14) is known as the *Hellinger distance*, which is the only instance in the family of α-divergences which is symmetric between both distributions.

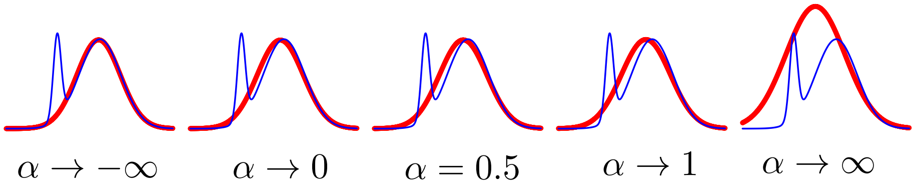

Figure 1: Changes on the approximate distribution q (in red) when trying to approximate it to the original distribution p (in blue) using different values for α in the α-divergence. When $\alpha \to -\infty$ the approximate distribution tries to cover a local mode of the target distribution (*exclusive distribution*). When $\alpha \to \infty$ the approximate distribution tries to cover the whole target distribution (*inclusive distribution*).

The value of the α parameter in the α-divergence has a strong impact on the inference results. To further understand its effect let us consider a toy problem in which we try to approximate a *slightly complex* distribution p with a simpler one, q. If we considered for example p as a bimodal distribution and q as a simple Gaussian distribution we could obtain results similar to those displayed in Figure 1 (reproduced from [20]). In this figure, the resulting unnormalized approximating distributions have different characteristics (the expression for the α-divergence can be generalized so that it can be evaluated on distributions that need not be normalized, see [20] for further details). First of all, in the limit of $\alpha \to -\infty$, q, here represented in red, tends to cover only the mode with the larger mass of the two present in p. By contrast, when $\alpha \to \infty$, q tends to cover the whole p distribution, overlaying the latter completely. This can be seen in terms of the form of the α-divergence. More precisely, for $\alpha \leq 0$, the α-divergence emphasizes q to be small whenever p is small (hence, it could be considered as *zero-forcing*). On the other hand, when $\alpha \geq 1$, it can be said that the divergence is inclusive, following the terminology in [21]. In this case, the divergence enforces $q > 0$ wherever $p > 0$, hence avoiding not having probability density in regions of the input space in which p takes large values.

In rest of the cases, α lays inside the interval $(0, 1)$. The behavior of q is intermediate between the two extreme possibilities that we have seen so far. In Figure 1 we can see that when $\alpha \to 0$ the q distribution is more centered in the main mode of p, whereas in $\alpha \to 1$ it begins to open to account for some of the mass of the secondary peak of p. This behavior also happens when the distributions considered are more complex than these ones, and therefore, one has to be careful when choosing α. In particular, the optimal value of α may depend on the task at hand and the particular model one is working with. As it has been pointed out before, when α is restricted to be in the interval $(0, 1)$ we can obtain two notable results at the extremes, $D_\alpha = \mathrm{KL}(q||p)$ for $\alpha \to 0$ and $D_\alpha = \mathrm{KL}(q||p)$ for $\alpha \to 1$. These two expressions are directly related to two of the main methods for approximate inference, Variational Inference [16] and Expectation Propagation [22], respectively.

Our method, AADM, will focus on minimizing the α-divergence for approximate inference instead of the typical KL-divergence. By adjusting the α parameter we expect to find the best compromise between zero-forcing and more inclusive approximate distributions. Recall that intermediate values of α in $(0, 1)$ are not zero-forcing, and therefore may try to capture multiple modes, but will ignore those modes that are *too far away* from the main mass of the distribution (*how far* will depend on the value of α). This extra flexibility is expected to better capture the properties of the posterior distribution.

4. Adversarial Alpha Divergence Minimization

So far we have seen that AVB is a flexible method for approximate inference that allows for the use of implicit models for the approximate distribution q_ϕ. If the implicit model is complex enough, AVB should be able to capture the features of the target distribution. However, AVB strongly relies on the KL divergence to enforce that the approximate distribution looks similar to the target distribution. In Section 3 we have pointed out that by employing a more general form of divergence one can obtain more flexible results, which depending on the task may mean a better balance between approximating a local mode of the posterior distribution (*exclusive distribution*) or having high probability density in all the regions of the input space in which the target distribution has high probability (*inclusive distribution*). The method proposed here is a generalization of AVB that allows for optimizing in an approximate way the α-divergence, instead of the KL divergence. By changing the α parameter one could obtain different approximate distributions $q(\mathbf{w})$ from the ones obtained in AVB. We refer to this new method as Adversarial Alpha Divergence Minimization (AADM). Our assumption here is that, if we are able to use values of α different from the ones that are used in AVB (*i.e.*, $\alpha \to 0$), we can perhaps obtain different approximating distributions that lead to more accurate predictive distributions in terms of different error metrics (*e.g.*, squared error, test log-likelihood, continuously ranked probability score, etc.), since it will allow the system to balance the importance assigned to mode-selecting and distribution covering.

As discussed earlier, when $\alpha \to 0$, the α-divergence recovers the KL divergence typical from VI and AVB, and when $\alpha \to 1$, the opposite KL divergence is restored (which is the one employed in other algorithms such as Expectation Propagation [22]). The mid range of values of α between 0 and 1 can be explored. Therefore, we will search for intermediate α values more suited for each learning task and performance metric.

To introduce the use of α-divergences in the context of AVB we need to modify the AVB objective function so that it accounts for this divergence between probability distributions. To do so we combine previous results from the literature. In particular, we briefly describe first black-box α, an extension of power expectation propagation, that is scalable and that allows for the approximate minimization of α-divergences [7]. Black-box α is, however, limited in the sense that it can only consider approximate distributions that belong to the exponential family, such as the Gaussian distribution. Therefore, we also introduce the reparametrization of the black-box α objective proposed in [23], which allows for more general approximate distributions. Finally, we will then derive the the objective function of the proposed approach, AADM. Unlike previous methods, (*e.g.*, power expectation propagation, black-box α, or the black-box α extension described in [23]) AADM will be able to consider implicit approximate distributions.

4.1. Black-box α-divergence Minimization

Black-box-α (BB-α) [7] is an improvement over the power expectation propagation method for approximate inference [24]. We will not explain here power expectation propagation (PEP) and refer the reader for the supplementary material for further details on that method. BB-α addresses some of the limitations of PEP like the $\mathcal{O}(N)$ memory space requirements, and also allows to make approximate inference on complicated probabilistic models [7]. To do so, BB-α maximizes a modified version of the objective of PEP. Namely,

$$\mathcal{L}(\phi) = \log Z_q - \log Z_{p(\mathbf{w})} + \frac{1}{\alpha} \sum_{i=1}^{N} \log \mathbb{E}_{q_\phi(\mathbf{w})} \left[\left(\frac{p(y_i|\mathbf{w}, \mathbf{x}_i)}{\tilde{f}(\mathbf{w})} \right)^{\alpha} \right] , \qquad (15)$$

where Z_q is the normalization constant of q_ϕ, $Z_{p(\mathbf{w})}$ is the normalization constant of the prior, ϕ are the parameters of $q(\mathbf{w})$, $p(y_i|\mathbf{w}, \mathbf{x}_i)$ is a likelihood factor and $\tilde{f}(\mathbf{w})$ is a global approximate likelihood factor that is replicated N times, one per each likelihood factor. This results in $q_\phi(\mathbf{w}) \propto \tilde{f}(\mathbf{w})^N p(\mathbf{w})$, which solves PEP's problem of having to store in memory the parameters of N approximate factors (as before, one per each likelihood factor). Furthermore, there is a one to one map between $\tilde{f}(\mathbf{w})$ and $q_\phi(\mathbf{w})$. This means that the max-min optimization problem of the PEP objective is transformed into just a standard maximization problem (w.r.t to the parameters of $q(\mathbf{w})$, ϕ), which can be solved using standard optimization techniques. Importantly, the expectations in (15) can be approximated via Monte Carlo sampling and the sum across the training data can be approximated using a mini-batch. The consequence is that BB-α scales to big datasets, as (15) can be optimized using stochastic techniques, and moreover, it can be applied to complicated probabilistic models (*e.g.*, Bayesian neural networks) in which the required expectations are intractable.

As in PEP, BB-α minimizes locally the α-divergence. In particular, it minimizes the sum of α-divergences between the approximate distribution q_ϕ and the tilted distributions, which are defined as $\hat{p}_i(\mathbf{w}) \propto \tilde{f}(\mathbf{w})^{N-1} p(y_i|\mathbf{w}, \mathbf{x}_i) p(\mathbf{w})$, for $i = 1, \ldots, N$. In general, it is expected that a local minimization of the α-divergence gives similar results to a global minimization, while being a much simpler problem, as indicated in [20]. When $\alpha \to 0$ (15) converges to the lower bound of VI in (5). When $\alpha = 1$, (15) is approximately equal to the objective optimized by Expectation Propagation [7]. A limitation of BB-α is, however, that the approximate distribution $q(\mathbf{w})$ is restricted to be inside the exponential family. This is because it must be written as the product of an approximate factor times the prior distribution. That is, $q_\phi(\mathbf{w}) \propto \tilde{f}(\mathbf{w})^N p(\mathbf{w})$. This is a major limitation that makes difficult using implicit models for $q(\mathbf{w})$.

4.2. Reparameterization of the Black-box-α Objective

In this section we introduce the reparametrization for the general expression of the BB-α objective that is suggested in [23] for approximate distributions with a closed form expression for the p.d.f. We combine this reparametrization with the trick of AVB to estimate the log-ratio between probability distributions. This will allow to approximately minimize α-divergences with flexible distributions $q(\mathbf{w})$ such as the ones resulting from implicit models. This will provide a complete definition of our method, AADM, extending the existing formulation of AVB. With this goal, we first consider the following alternative expression for the BB-α objective that is suggested in [23]:

$$\mathcal{L}_\alpha(\phi) = \frac{1}{\alpha}\sum_{i=1}^{N} \log \mathbb{E}_{q_\phi(\mathbf{w})}\left[\left(\frac{p(y_i|\mathbf{x}_i,\mathbf{w})p(\mathbf{w})^{1/N}}{q_\phi(\mathbf{w})^{1/N}}\right)^{\alpha}\right] . \tag{16}$$

In this expression we observe that the hypothesis that $q_\phi(\mathbf{w}) \propto \tilde{f}^N(\mathbf{w})p(\mathbf{w})$ is not required anymore (both Z_q and $\tilde{f}(\mathbf{w})$ are removed from the expression) and $q(\mathbf{w})$ can be an arbitrary distribution. It is possible to show that (16) and (15) become equivalent if $q(\mathbf{w})$ belongs to the exponential family [23]. However, this expression requires the evaluation of the density $q_\phi(\mathbf{w})$, which in practice may be hard to compute.

To overcome this previous limitation we can follow similar steps to [23], reparameterizing (16) using the so-called *cavity distribution*. That is, the distribution given by the ratio $q_\phi/\tilde{f}^\alpha$. If $\tilde{q}_\phi(\mathbf{w})$ denotes a free-form cavity distribution, the posterior approximation q_ϕ is given by:

$$q_\phi(\mathbf{w}) = \frac{1}{Z_q}\tilde{q}_\phi(\mathbf{w})\left(\frac{\tilde{q}_\phi(\mathbf{w})}{p(\mathbf{w})}\right)^{\frac{\alpha}{N-\alpha}} \tag{17}$$

where we assume $Z_q < +\infty$ is the normalizing constant to make $q(\mathbf{w})$ a valid distribution. When $\alpha/N \to 0$ we have that $q \to \tilde{q}$ (and $Z_q \to 1$ by assumption), and this is the case either if we choose $\alpha \to 0$, or when N is sufficiently large (i.e. $N \to +\infty$), see [23]. We rewrite now (16) in terms of $\tilde{q}$ rather than $q(\mathbf{w})$, as in [23]:

$$\begin{aligned}
\mathcal{L}_\alpha(\phi) &= \frac{1}{\alpha}\sum_{i=1}^{N}\log\int\left(\frac{1}{Z_q}\tilde{q}_\phi(\mathbf{w})\left(\frac{\tilde{q}_\phi(\mathbf{w})}{p(\mathbf{w})}\right)^{\frac{\alpha}{N-\alpha}}\right)^{1-\frac{\alpha}{N}} p(\mathbf{w})^{\frac{\alpha}{N}} p(y_i|\mathbf{w},\mathbf{x}_i)^\alpha d\mathbf{w} \\
&= -\frac{N}{\alpha}\left(1-\frac{\alpha}{N}\right)\log\int \tilde{q}_\phi(\mathbf{w})\left(\frac{\tilde{q}_\phi(\mathbf{w})}{p(\mathbf{w})}\right)^{\frac{\alpha}{N-\alpha}} d\mathbf{w} \\
&\quad + \frac{1}{\alpha}\sum_{i=1}^{N}\log\mathbb{E}_{\tilde{q}_\phi(\mathbf{w})}\left[p(y_i|\mathbf{x}_i,\mathbf{w})^\alpha\right] \\
&= \frac{1}{\alpha}\sum_{i=1}^{N}\log\mathbb{E}_{\tilde{q}_\phi(\mathbf{w})}\left[p(y_i|\mathbf{x}_i,\mathbf{w})^\alpha\right] - \mathrm{R}_\beta[\tilde{q}|p] ,
\end{aligned} \tag{18}$$

where $\beta = N/(N-\alpha)$ and $\mathrm{R}_\beta[\tilde{q}|p]$ represents the *Rényi divergence* of order β [26], which is defined as

$$\mathrm{R}_\beta[q|p] = \frac{1}{\beta - 1} \log \int \tilde{q}(\mathbf{w})^\beta p(\mathbf{w})^{1-\beta} d\mathbf{w}. \tag{19}$$

Importantly, when $\alpha/N \to 0$ we recover $q \to \tilde{q}$ and $\mathcal{L}_\alpha(\phi)$ converges to the objective of VI. Also, we have that $\mathrm{R}_\beta[\tilde{q}|p] \to \mathrm{KL}[\tilde{q}||p] = \mathrm{KL}[q||p]$ if $\mathrm{R}_\beta[\tilde{q}|p] < +\infty$ (which is true assuming $Z_q < +\infty$ and $\alpha/N \to 0$). Therefore, following [23], when this quotient tends to zero, we can make further approximations for the BB-α energy function, as described in (16), finally obtaining

$$\mathcal{L}_\alpha(\phi) \approx \frac{1}{\alpha} \sum_{i=1}^{N} \log \mathbb{E}_{q_\phi(\mathbf{w})}[p(y_i|\mathbf{x}_i, \mathbf{w})^\alpha] - \mathrm{KL}[q_\phi(\mathbf{w})||p(\mathbf{w})] \,. \tag{20}$$

This will be the objective function that we will maximize in our approach. Note that the expectations in (20) can be estimated via Monte Carlo sampling. In particular, $\log \mathbb{E}_{q_\phi(\mathbf{w})}[p(y_i|\mathbf{x}_i,\mathbf{w})^\alpha] \approx \log[K^{-1} \sum_{k=1}^{K} p(y_i|\mathbf{x}_i, \mathbf{w}_k)^\alpha]$, for K samples of $\mathbf{w}$ drawn from q_ϕ. Of course, this estimate is biased, as a consequence of the non-linearity of the $\log(\cdot)$ function, however, the bias can be controlled with K. Furthermore, we expect a similar behavior as in standard BB-α, in which the bias has been shown to be very small even for $K = 10$ samples. See [7] for further details.

The objective in (20) has been obtained under some conditions that need not be true in practice, *e.g.* the quotient $\alpha/N \to 0$ (*i.e.*, either α is small, N is sufficiently large or a combination of both). Nevertheless, it is much simpler to estimate and maximize than the objective in (16). It is also similar to the objective functions found in the deep learning bibliography (*i.e.*, a loss function plus some regularizer, such as the KL divergence), but it still maintains the qualities of an approximate Bayesian inference algorithm. Importantly, (20) allows for implicit models for q_ϕ. The only term that is difficult to approximate is $\mathrm{KL}[q_\phi(\mathbf{w})||p(\mathbf{w})]$. However, the approach described in Section 2.2 for AVB can be used here for that purpose. We can simply use an independent classifier to estimate the log-ratio between $p(\mathbf{w})$ and $q_\phi(\mathbf{w})$, as in AVB. This enables using implicit distributions when maximizing the objective in (20).

By changing the α parameter of the method we will be able to interpolate between AVB ($\alpha \to 0$) and an EP-like algorithm ($\alpha = 1$). Note that when $\alpha \to 0$, (20) is expected to focus on reducing the training error since the factor $\alpha^{-1} \log \mathbb{E}_{q_\phi(\mathbf{w})}[p(y_i|\mathbf{x}_i, \mathbf{w})^\alpha]$ will converge to $\mathbb{E}_{q_\phi(\mathbf{w})}[\log p(y_i|\mathbf{x}_i, \mathbf{w})]$, with $p(y_i|\mathbf{x}_i, \mathbf{w})$ typically a Gaussian distribution with mean given by the output of the neural network and noise variance σ^2. By contrast, when $\alpha = 1$, (20) will be expected to focus more on the training log-likelihood. Intermediate values of α will trade-off between these two tasks, which may lead to better generalization properties of the predictive distribution.

The specific details of the structure of the proposed approach, AADM, are analogous to the ones described for AVB [10]. The structure of AADM can be divided into three main components: An implicit model for q_ϕ, which takes as input Gaussian noise and outputs neural network weight samples $\mathbf{w}$ from the approximated weights posterior distribution (*i.e.*, the generator network); a discriminator, which estimates the KL term present in (20) as done in [10]; and finally the main network, that uses the samples of the weights generated previously to evaluate the factor $p(y_i|\mathbf{x}_i, \mathbf{w})$. The whole system is optimized altogether. Furthermore, any potential hyper-parameter (*e.g.*, the prior variance σ_0^2 or the output noise variance σ^2) is tuned simply by maximizing the objective in (20).

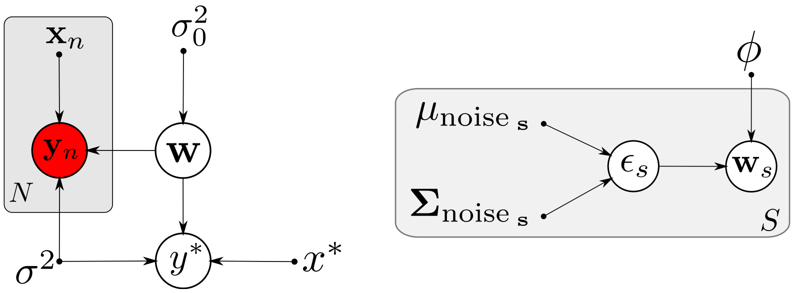

Figure 2: Graphical models for AADM. (left) - Assumed probabilistic graphical model for the observed data. Point-like vertices denote deterministic variables and circular ones indicate random variables, which can either be *observed* (red) or *unobserved* (white). (right) - Probabilistic graphical model of the implicit distribution q used to approximate the posterior. A source of S samples of Gaussian noise (we assume independence) with mean $\boldsymbol{\mu}_{\text{noise}}$ and variances $\boldsymbol{\Sigma}_{\text{noise}}$ is let through through a deep neural network with parameters ϕ to generate S samples of the weights of the main neural network. Best seen in color.

Figure 2 (left) shows the main probabilistic graphical model corresponding to the observed data. Point-like vertices indicate deterministic variables. Circular vertices denote random variables, which can be *observed* (red) or *unobserved* (white). Figure 2 (right) shows the probabilistic graphical model of the implicit approximate posterior distribution. In this case, we generate S samples of Gaussian noise in the form of $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{\mu}_{\text{noise}}, \boldsymbol{\Sigma}_{\text{noise}})$, with $\boldsymbol{\Sigma}_{\text{noise}}$ a diagonal matrix. These samples are passed through a deep neural network with weights ϕ to obtain S samples for the weights of the main neural network. These weights are then used in the main network, shown in Figure 2 (left), to estimate the predictive distribution during training and testing.

As a last remark concerning the implementation of the proposed method, we have also included as trainable parameters both the mean and variances of the Gaussian noise which is used as input in the generator network (the implicit model for the weights, $q_\phi(\mathbf{w})$, in (6)). This allows for a more expressive implicit model for $q_\phi(\mathbf{w})$, since it increases its flexibility by enabling the tuning of the broad parameters that control its input. Using this in combination with the approximate minimization of α-divergences, the proposed method AADM is expected to reproduce to a higher degree of accuracy the original posterior distribution of the model parameters (neural network weights). Our hypothesis is that this will lead to more accurate predictive distributions.

Finally, the proposed method AADM may suffer from convergence to bad local optima. This may happen as a consequence of the strong regularization effect of the term $\mathrm{KL}[q_\phi(\mathbf{w})||p(\mathbf{w})]$ at the beginning of the training process. To alleviate this problem and obtain better results, we have considered also the approach suggested in [27] that consists in adding an extra annealing parameter β that penalizes the KL term. This parameter takes value 0 at the beginning and progressively, after each epoch, it increases until it takes value 1. See the supplementary material for further details.

5. Related Work

Obtaining uncertainty estimates associated to the predictions of machine learning algorithms is a widely spread problem. The problem of approximating the posterior has been addressed either by sampling-based methods or by optimization-based methods [14]. In the former case, the posterior distribution is approximated by drawing samples from the exact posterior to then use them for inference and prediction. With this goal, a Markov chain is run, whose stationary distribution coincides with the target distribution. On the other hand, optimization-based methods introduce an approximate distribution $q(\mathbf{w})$ whose parameters are adjusted to match the exact posterior through the optimization of a certain objective.

Each of the approaches described has advantages and disadvantages. First, sampling methods can be unbiased only asymptotically, and moreover they can be highly computationally expensive since the Markov chain has to be run for long time in practice. Similarly, optimization-based techniques are usually limited by the definition of the approximating distribution, which is often parametric, and therefore they may lack expressiveness. Two examples of these methods are Markov chain Monte Carlo (MCMC) in the case of sampling-based methods [28, 29, 5], and variational inference (VI) or expectation propagation (EP) in the case of optimization-based methods [22, 30, 8, 16, 31]. The method proposed here alleviates some of the problems of these two techniques. Specifically, it allows for flexible approximate distributions and it also scales to large datasets, whereas in some of these cases, large datasets can be a burden to deal with [32].

Most modern techniques for approximate inference take advantage of the speed of optimization-based methods and try to preserve the flexibility of sampling-based methods with the goal of obtaining the best results possible in terms of computational cost and accuracy of the approximation. There are, however, many different ways of combining both approaches, which is showcased by the wide variety of methods proposed. In this section we review some of them. Nevertheless, almost all of them rely on optimizing the KL divergence between $q(\mathbf{w})$ and the target distribution. Our approach is more general and can minimize, in an approximate way, the α-divergence, which as we pointed out before, includes the KL divergence as a particular case, as well as other divergences (*e.g.* the Hellinger distance).

In [33] it is described how to estimate the gradient of the VI objective when using an implicit model for the approximate distribution $q(\mathbf{w})$. This gradient can then be used to maximize the objective. The method proposed there combines Markov chain Monte Carlo methods and VI. While this seems promising, its implementation is complicated since it relies on running an inner Markov chain inside the optimization process of the approximate distribution $q(\mathbf{w})$. Moreover, the parameters of the Markov chain also may need to be adjusted, depending on the probabilistic model used in practice.

Another approach for flexible approximate distributions $q(\mathbf{w})$ within the context of VI is normalizing flows (NF) [9]. In NF one starts with a simple parametric approximate distribution $q(\mathbf{w})$ whose samples are modified using parametric non-linear invertible transformations that are carefully chosen. This results in a p.d.f. of the resulting distribution that can be evaluated in closed form, avoiding the problems arising from the use of implicit models for $q(\mathbf{w})$. Nevertheless, the family of transformations that can be used is limited to invertible transformations, which may constrain the flexibility of the approximate distribution $q(\mathbf{w})$.

Stein Variational Gradient Descent, proposed in [11], is a general VI method that consists in transforming a set of *particles* to match the exact posterior distribution. The results obtained are shown to be competitive with other state-of-the-art methods, but the main drawback here is that there is a computational bottleneck on the number of particles that need to be stored to accurately represent the posterior distribution. More precisely, this method lacks a way to generate samples from the approximate distribution $q(\mathbf{w})$. The number of samples is fixed initially, and these are optimized by the method.

The work in [12] combines VI and MCMC methods to obtain flexible approximate posterior distributions. The key concept is to use a Markov chain as the approximate distribution $q(\mathbf{w})$ in VI. The parameters of this chain can then be adjusted to match the target distribution in terms of the KL divergence as close as possible. This is an interesting idea, but it is also limited by the difficulty of evaluating the p.d.f. of the approximate distribution. This is solved in [12] by learning a backward model, that infers the p.d.f. of the initial state of the Markov chain given the generated samples. Learning this backward model accurately is a complex task and several simplifications are introduced that may affect the results.

Another approach used for approximate inference in the context of Bayesian neural networks is Probabilistic Back-propagation [6]. This method computes a forward propagation of probabilities through the neural network to then do back-propagation of the gradients. Although it has been proven to be a fast approach with high performance, it is limited by the expressiveness of the posterior approximation. In particular, the approximate distribution is restricted to be Gaussian. This means that this method will suffer from strong approximation bias. The same applies to a standard application of VI in the context of Bayesian neural networks [8, 17].

The minimization of α-divergences in the context of Bayesian neural networks has also been addressed in [6]. In that work it is described Black-box-α, a method for approximate inference that allows for very complex probabilistic models and that is efficient and allows for big datasets. The main limitation is, however, that the approximate distribution $q(\mathbf{w})$ must belong to the exponential family. That is, the approximate distribution has to be Gaussian, and hence this method will also suffer from approximation bias. Therefore, Black-box-α is expected to be sub-optimal when compared to the method proposed in this paper, which allows for implicit models in the approximate distribution $q(\mathbf{w})$.

The minimization of α-divergences has also been explored in the context of dropout in [23]. That work considers the same objective as the one optimized by our approach in Section 4.2. The difference is that the approximate distribution considered by the authors of that work is limited to the approximate posterior distribution of dropout. This distribution is given by the mixture of two points of probability mass, *i.e.*, two delta functions, one of which is located at the origin [34]. The flexibility of this approximate distribution is therefore very limited. By contrast, the method we propose allows for implicit approximate distributions $q(\mathbf{w})$ and is expected to give superior results.

Finally, a closely related method to ours is the one described in [10]. This method, Adversarial Variational Bayes (AVB), allows to carry out Variational Inference with implicit models as the approximate distribution $q(\mathbf{w})$. For this, in that work it is proposed to train a discriminator whose output can be used to estimate the KL divergence between the approximate distribution $q(\mathbf{w})$ and the prior. This technique has also been considered in other works [13, 18, 14]. A limitation of AVB is that the method is restricted to minimize the KL divergence between the approximate and the target distribution. Our approach, by contrast, can optimize the more general α-divergence, which includes the KL divergence as a particular case. Therefore, by changing the α parameter our method can potentially obtain better results than AVB. This hypothesis is confirmed by the experiments in the next section.

6. Experiments

To analyze and evaluate the performance of the proposed approach, *i.e.*, Adversarial α-divergence Minimization (AADM), we have carried out extensive experiments, both in synthetic data and on common UCI datasets [35]. Furthermore, we have compared results with previously existing methods such as VI, using a factorizing Gaussian as the approximate distribution, and AVB. AADM should give the similar results as AVB for $\alpha \to 0$. In these experiments we have also analyzed performance versus computational cost of each method on larger datasets with up to 2 million data points.

The method AADM employed in our experiments consists in the previously described three-network system. In particular, the structure we have considered for AADM (and also AVB), if not stated otherwise, is the following one: The generator network takes as an input a 100-dimensional Gaussian noise sample, with adjustable mean and diagonal covariance parameters, and passes it through 2 layers of 50 non-linear units each, outputting a sample of the weights $\mathbf{w}$. We generate 10 samples for the weights when training, and 50 samples to approximate the predictive distribution when testing. Similarly, the discriminator takes these samples of the weights (as well as samples from the auxiliary distribution from the Adaptive Contrast) and passes them through 2 layers of 50 non-linear units each to compute $T_\omega(\mathbf{w})$. Finally, the main network (*i.e.*, the model whose weights we are inferring) also consists of a 2 layer system with 50 units per layer as well. This network uses the sampled weights and the original data as input to estimate the AADM objective $\mathcal{L}_\alpha(\phi)$. Note that although the network size employed in our experiments is small, it is similar to the network size considered in recent related works [6, 23].

The number of training epochs and the presence (or absence) of a warm-up period depends on the dataset being used, and therefore is specified in each experiment. All non-linear units are leaky ReLU units. The code implementing the proposed approach is available online at `https://github.com/simonrsantana/AADM`. All methods have been trained using stochastic optimization via ADAM [36]. The learning rate for updating the parameters of the discriminator is set to the default value in ADAM, *i.e.*, 10^{-3}. The learning rate for updating the implicit model for q_ϕ (*i.e.*, the generator) and the model hyper-parameters (which includes the variance of the output noise and the prior) is set to 10^{-4}. Apart from this, we use the default parameter values in ADAM. The mini-batch size used is described in each experiment.

In our experiments we have evaluated 3 different performance metrics. The test log-likelihood is evaluated, as well as a metric concerning the error of the predictions (*i.e.* RMSE, in regression problems, and classification error, in binary and multi-class classification problems). We have also used a third metric in each case as well, with the goal of measuring the quality of the predictive distribution, as an alternative to the test log-likelihood. More precisely, we have used strictly proper scoring rules defined in [15]. For the regression experiments we have employed the *Continuous Ranked Probability Score* (CRPS), which has a close-form expression described in [37]. On the other hand, in classification problems we have used the *Brier score*, both for binary and multi-class problems. The CRPS is the squared distance between the c.d.f. of the empirical distribution of the target variable and the c.d.f. of the predictive distribution. The Brier score is simply the squared distance between the vector of predictive probabilities for each class and a vector with a one-hot encoding of the observed class. For further information about these metrics, please see [15]. In general, the smaller their value, the better, and a metric value equal to zero means a perfect predictive distribution.

6.1. Synthetic Experiments

To illustrate the features of the predictive distribution that the proposed approach AADM can capture, we evaluate this method on two simple regression problems extracted from [38]. More precisely, we generate two different *toy datasets*. The first one involving a heteroscedastic predictive distribution, and the second one involving a bimodal predictive distribution.

The structure of the system employed is the one described previously. We train this system for 3000 epochs, using the first 500 epochs as the warm-up period. We repeat the experiments for different values of alpha in the $(0, 1]$. The first dataset is generated taking x uniformly distributed in the interval $[-4, 4]$ and y is obtained as $y = 7 \sin x + 3|\cos(x/2)|\epsilon$, where ϵ is normally-distributed and independent of x, *i.e.*, $\epsilon \sim \mathcal{N}(0, 1)$. Note that this dataset involves input dependent noise. The second dataset uses x uniformly distributed in the interval$[-2, 2]$ and $y = 10 \sin x + \epsilon$ with probability 0.5 and $y = 10 \cos x + \epsilon$ otherwise. The distribution of ϵ is the same as in the first dataset. note that this other dataset involves a bimodal predictive distribution. We use 1000 data instances for training and the mini-batch size is set to 10.

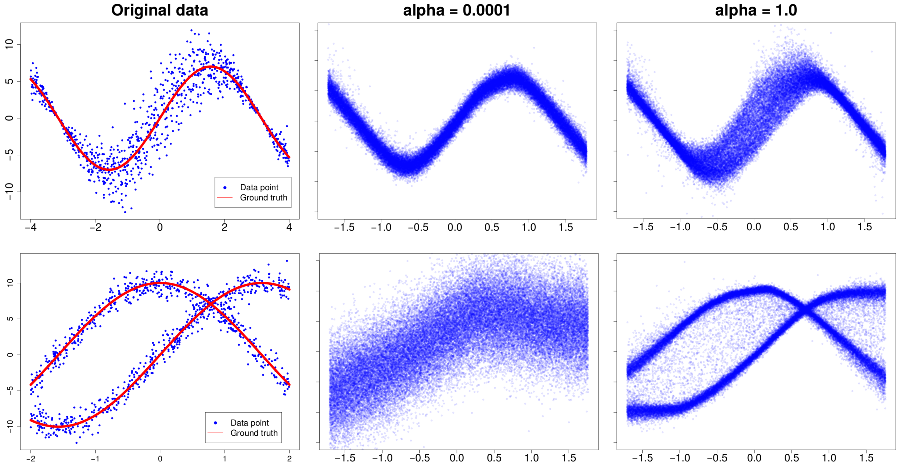

Figure 3: Results for the toy problems. The blue points on the left represent the original training data and the ground truth (red lines). In the middle, normalized predictions generated with $\alpha \approx 0$ (i.e. regular AVB), and in the right side are the normalized predictions with $\alpha = 1.0$.

The results obtained in the synthetic problems described are represented in Figure 3. The figures on at the top correspond to the problem involving the heteroscedastic noise and the bottom ones to the problem with a bimodal predictive distribution. On the left of the figure we show the original data we used to train AADM. In these plots, the red lines represent the *ground*

Table 1: Test log-likelihood, RMSE and CRPS for AADM with $\alpha = 10^{-4}$ and $\alpha = 1.0$ in both toy experiments.

	Bimodal			**Heteroscedastic**		
$\boldsymbol{\alpha}$	**L-L**	**RMSE**	**CRPS**	**L-L**	**RMSE**	**CRPS**
10^{-4}	-3.05	5.10	3.28	-2.10	1.91	1.07
1.0	-2.23	5.09	2.65	-1.91	1.94	0.97

truth for each dataset and the blue points are the actual samples we used as training data. The middle and right columns show normalized samples from the predictive distribution of a neural network trained using AADM, for $\alpha = 10^{-4}$ and $\alpha = 1$, respectively. The results obtained for $\alpha = 10^{-4}$ are expected to be equal to those of AVB. A low value for α is unable to capture the complex structure of predictive distribution for the target variable, ignoring features such as the heteroscedastic noise in the first task, and the bimodality of the predictive distribution in the second task. However, both of these features are captured with accuracy when α is higher, as illustrated by the results obtained when $\alpha = 1$.

As expected, choosing one value of α or another in AADM significantly changes the results obtained. In particular, when $\alpha = 10^{-4}$ the predictive distribution focuses more on minimizing the squared error and less on the log-likelihood of the data. By contrast, when $\alpha = 1.0$, the predictive distribution plays a closer attention to the log-likelihood of the data, and can hence obtain a more accurate predictive distribution. As shown in Table 1, although the squared error obtained when $\alpha = 10^{-4}$ and $\alpha = 1.0$ is very similar, the test log-likelihood obtained when $\alpha = 1.0$ is much better, which indicates that this value of α produces more accurate predictive distributions. Moreover, the CRPS values also improve when α is 1.0 rather than 10^{-4}. Note that the squared error only measures the expected squared deviation from target value. On the other hand, the test log-likelihood and the CRPS, measure the overall quality of the predictive distribution, taking into account, for example, features such as multiple-modes, heavy-tails or skewness.

Table 2: Characteristics of the UCI datasets used in the experiments.

Dataset	**Instances**	**Attributes**	**Epochs**
Boston	506	13	2000
Concrete	1,030	8	2000
Energy Efficiency	768	8	2000
Kin8nm	8,192	8	400
Naval	11,934	16	400
Combined Cycle Power Plant	9,568	4	250
Wine	1,599	11	2000
Yatch	308	6	2000

Finally, other values of α give similar results (not shown here). In particular, for $\alpha < 0.5$ similar results to those of $\alpha = 10^{-4}$ are obtained. By contrast, when $\alpha > 0.5$ similar results to those of $\alpha = 1.0$ are obtained (only if the training procedure is carried out carefully to avoid bad local optima).

6.2. Experiments on UCI Datasets

To analyze in more detail the results of the proposed method, AADM, we have considered eight UCI datasets [35] that are widely spread for regression [6]. The characteristics of these datasets are displayed in Table 2. Each dataset has a different size, and in order to train the different methods until convergence we have employed a different number of epochs in each case. The number of epochs selected is presented finally in Table 2. Note that, even though there are differences in the epochs employed for training, all of the datasets share the same model structure, which is the general one described at the beginning of this section. In all these experiments we employ the first 10% of the total training epochs for *warming-up* before the KL term is completely turned on as in [27]. Moreover, the batch size is set to be 10 data points, and sampling-wise, we perform 10 samples in the training procedure and 100 for testing. We split the datasets in a 90%-10% for training/testing. The results reported are averages over 20 different random splits of the datasets into training and testing.

We compare the results of AADM with VI using a factorizing Gaussian as the posterior approximation and with regular AVB (which should be the same as our algorithm when $\alpha \to 0$). For all methods we employ the same two-layered system with 50 units per layer. To make fair comparisons we also perform the same warm-up period for both AVB and VI as we use in our method. Therefore only after the first 10% of the total number of epochs, the KL term is completely activated in the objective function.

The average performance of each method on each dataset, in terms of the test log-likelihood, is displayed Figure 4. In this case, the higher, the better. The test log-likelihood measures the overall quality of the predictive distribution, taking into account, for example, features such as multiple-modes, heavy-tails or skewness. We observe that values of α that are different from 0 usually outperform both regular AVB and VI in terms of this metric (the higher the values the better). From these figures, it seems that higher values of α often lead to better predictive distributions it terms of the test log-likelihood, probably as a consequence of being able to better recover the real posterior distribution. The values obtained are similar and often better than those of other state of the art methods [6]. Each of the values shown represent the mean performance of a certain method across the 20 different splits of each dataset, which are averaged afterwards here. Importantly, we observe that standard VI is almost always outperformed by the two techniques that allow for implicit models in the posterior approximation $q(\mathbf{w})$. Namely, AVB and AADM. This points out the benefits of using an implicit model for the approximate distribution $q(\mathbf{w})$. Moreover, AVG and AADM give almost the same results when $\alpha \approx 0$, which confirms the correctness of our implementation.

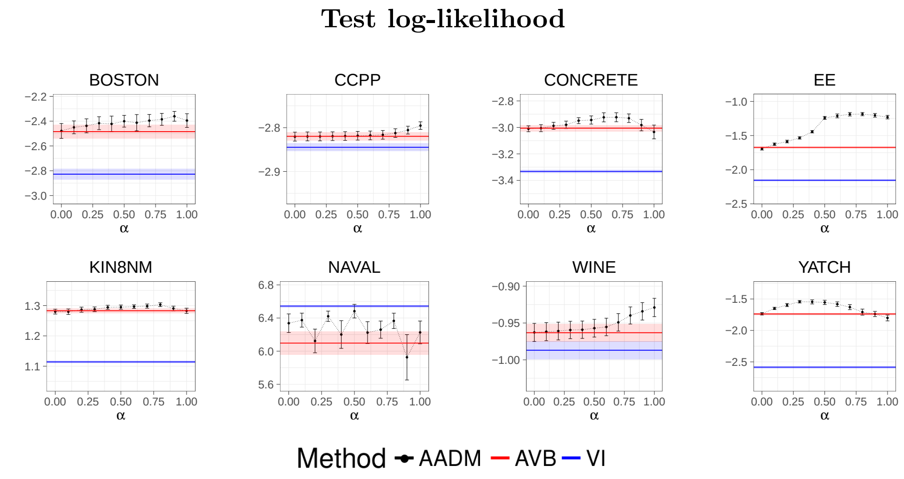

Figure 4: Average results in terms of the test log-likelihood for the different UCI datasets and methods compared (higher values are better). Black represents the performance for our method, AADM, for different values of α. Red is the performance of AVB. VI is presented in blue. Best seen in color.

The average results obtained for each method on each dataset, in terms of the root mean squared error (RMSE) are displayed in Figure 5. Note that the root mean squared error only measures the expected deviation from the target value and it may ignore if the model captures accurately the distribution of the target value. We can see that the proposed approach, AADM, also obtains better results than VI. In this case, nonetheless, increasing α values do not actually improve much over the basic results of AVB, and in general we can see that lower values for α are actually better for obtaining a good performance in terms of this metric (here, the lower the values the better the performance). This seems to indicate that one should choose a value for α that is different, depending on the metric they are most interested in. These results are consistent in the sense that, as pointed out previously, values of α close to zero actually lead to the objective that is optimized in AVB and VI, which pays more attention to the training RMSE. By contrast, values closer to 1.0 result in an objective function that is more closely related to the log-likelihood of the training data.

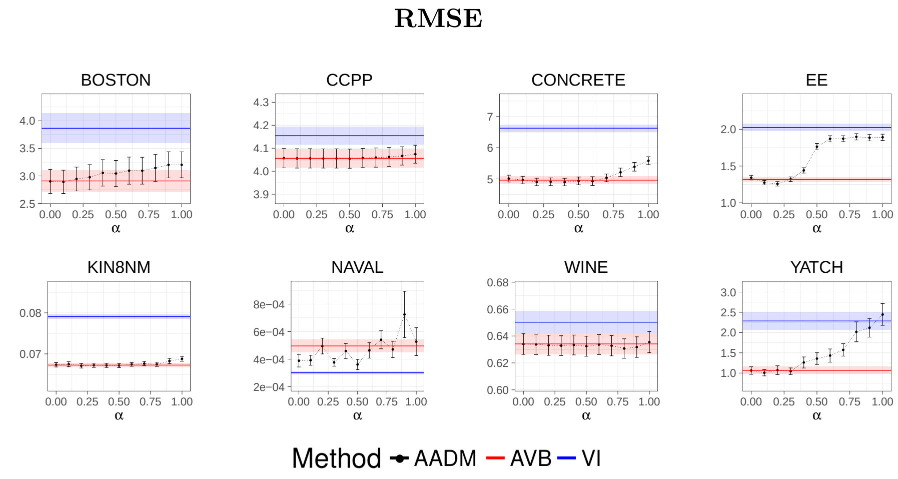

Figure 5: Average results in terms of the root mean squared error for the different UCI datasets and methods compared (lower values are better). Black represents AADM for different values of α, red is AVB, and VI is presented in blue. Best seen in color.

We also report the performance of each method in terms of CRPS metric (lower values are better) in Figure 6. The results obtained are similar to those obtained in terms of the test log-likelihood. This is the expected behavior since the CRPS metric also evaluates quality of the predictive distribution for the test data and it should be correlated with the test log-likelihood. In particular, values of α different from 0 are expected to give more accurate predictive distributions. This is confirmed by the results. It seems that the CRPS metric improves in general when α increases. However, there are few exceptions, such as those of the *Yatch* and *Naval* datasets.

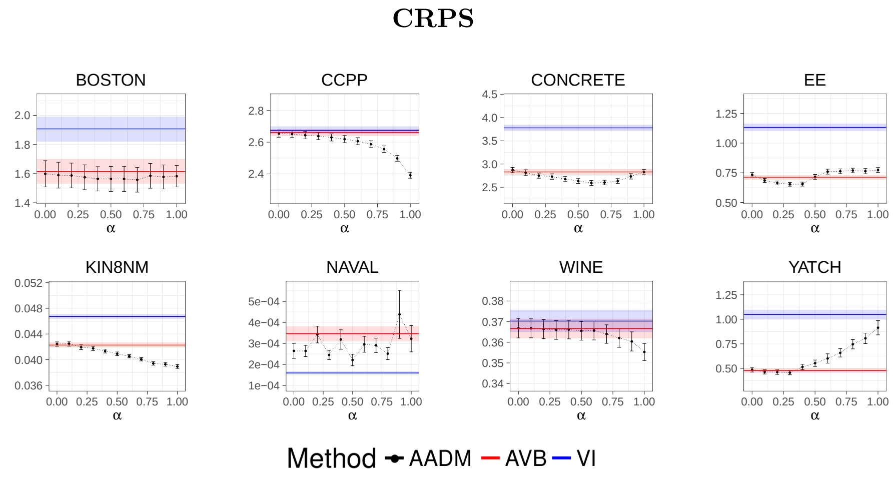

Figure 6: Average results in terms of the continuous ranked probability score (CRPS) for the different UCI datasets and methods compared (lower values are better). Again, black here is AADM, red represents AVB and blue, VI. Best seen in color.

6.2.1. Average Rank Results on the UCI Datasets

To get an overall idea about the performance of AADM, for each value of α, on the previous experiments we have proceeded as follows: We have ranked the performance AADM for each α value (*i.e.*, rank 1 means that value of α gives the best result, rank 2 means that it gives the second best results, etc.). Then, we have computed the average rank over all the train / test splits of the datasets, and have calculated the standard deviation in each case. Figure 7 shows the results obtained for the RMSE, test log-likelihood and CRPS.

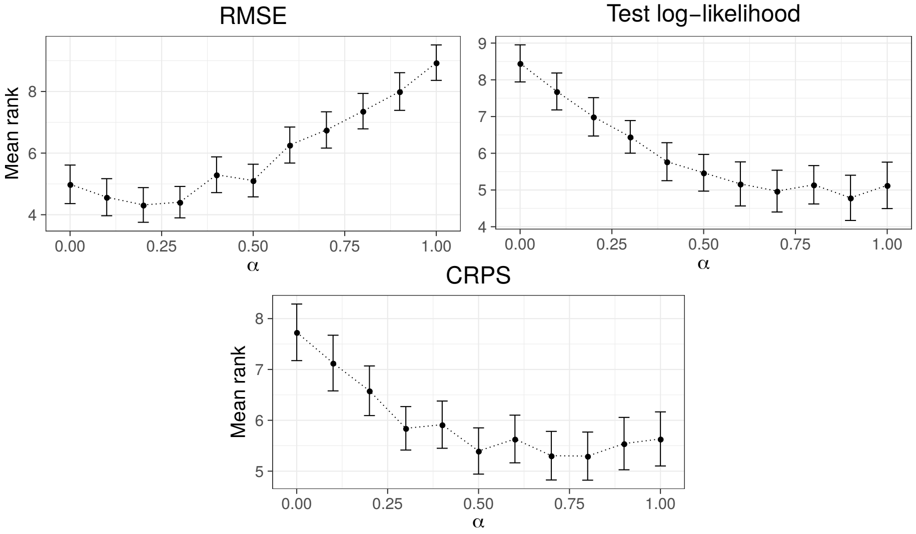

Figure 7: Average rank (the lower the better) for AADM and each value of α in terms of the RMSE (first row, left), test log-likelihood (first row, right) and CRPS (second row) across all the UCI datasets and splits.

The results obtained are displayed in Figure 7. This figure confirms that the intermediate values of alpha usually present a better performance than the extremes (*i.e.*, $\alpha \approx 0$ or $\alpha = 1$), for either the RMSE, the test log-likelihood metrics and the CRPS). Furthermore, both for the test log-likelihood and the CRPS, higher values of α provide better results, which means that these values of α provide more accurate predictive distributions. This is expected to be related to a better posterior approximation. In spite of this, lower values of α tend to perform better in terms of the RMSE (although the best results are still obtained when $\alpha > 0$). Again, this can be explained by paying attention to the form of the objective function that is maximized in both extremes, *i.e.*, for $\alpha \to 0$ and for $\alpha = 1$. Recall that the the VI objective is recovered when $\alpha \to 0$. This objective gives higher importance to the squared error since $\log p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w})$ is precisely the squared error. By contrast, a similar objective function to the one used by expectation propagation is obtained when $\alpha = 1$. This objective includes terms that involve the log-likelihood of the training data. That is,

$\log \mathbb{E}_q[p(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w})]$. The main conclusion from this analysis is that the optimal value for α depends on the metric we are considering, and that intermediate values of α, different from 0 or 1 can lead to better results.

A question that may arise at this point is how to choose the α value for a given task. In a broad sense, the optimal choice would strongly depend on the performance metric we are interested in. More precisely, as we have seen in the results of Figure 7, lower values of α tend to produce better predictive distributions in terms of the squared error, while higher values of α lead to better the predictive distributions in terms of both the test log-likelihood and the CRPS. Moreover, at the very least, $\alpha > 0$ improves the general performance of pre-existing methods, as can be concluded from Figures 4, 5 and 6. Our recommendation is to set α close to 0 if one is interested in low squared error and to choose α close to 1 if one is interested in capturing more general features of predictive distribution. In practice, however, one should carry out a model validation procedure (*e.g.* using cross validation) to test each value of α.

6.3. Binary and Multi-class Classification

We have also evaluated AADM in several binary classification tasks, and on two multi-class problems. Namely, the MNIST and CIFAR-10 datasets. The results of these experiments are found in the supplementary material . In those experiments, however, the differences among all the methods are very small. In spite of this, AADM has shown to be competitive providing slightly better results than those of VI and similar results to those of AVB.

6.4. Experiments on Big Datasets

To evaluate the performance of the proposed method on large datasets, we have carried out additional experiments considering two datasets: *Airlines Delay*, and *Year Prediction MSD*. Airlines Delay contains information about all commercial flights in the USA from January 2008 to April 2008 [39]. The task of interest is to predict the delay in minutes of a flight based on 8 attributes: age of the aircraft, distance that needs to be covered, air-time, departure time, arrival time, day of the week, day of the month and month. This is hence a very noisy dataset. After removing instances with missing values, $2,127,068$ instances remain. From these, $10,000$ are used for testing and the rest are used for training. *Year Prediction MSD* is publicly accessible on the UCI repository [35]. This dataset has $515,345$ data instances and 90 attributes. Again, we use $10,000$ instances for testing and the rest of the data are used for training. In these experiments the mini-batch size has been set to 100 and we have not used the warm-up annealing scheme that deactivates the KL term in the objective of each method during the initial training iterations. For each method, we measured the performance in the test set, in terms of the RMSE, the test log-likelihood and the CRPS as a function of the training time.

The results obtained for each method on the Airlines dataset are displayed in Figure 8. In this figure dashed lines represent other methods, the black being AVB and the blue VI. Solid lines represent our method, AADM, for different values of alpha. The figure shows that AADM obtains better results than AVB and VI in terms of the test log-likelihood and CRPS when α approaches 1. When α is closer to 0, AADM, gives similar results to those of AVB and VI in the long term. The performance of our method w.r.t. the computational time is comparable to that of AVB. In terms of RMSE, however, large values of α seem to exhibit a more unstable behavior and in general give worse results. This is probably a consequence of this dataset being very noisy.

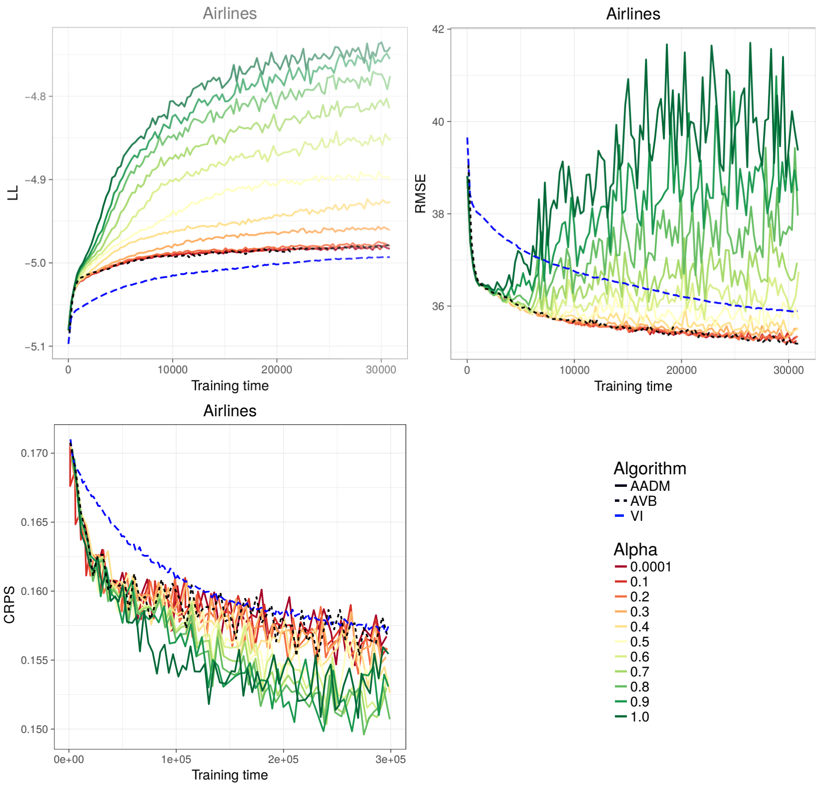

Figure 8: Performance as a function of the computational time in the Airlines dataset for each method. We report both in test log-likelihood (top-left), RMSE (top-right) and CRPS (bottom). The dashed blue line corresponds to the method VI, the dashed black line to AVB, and other solid lines represent our method, AADM, for different values of alpha. Best seen in color.

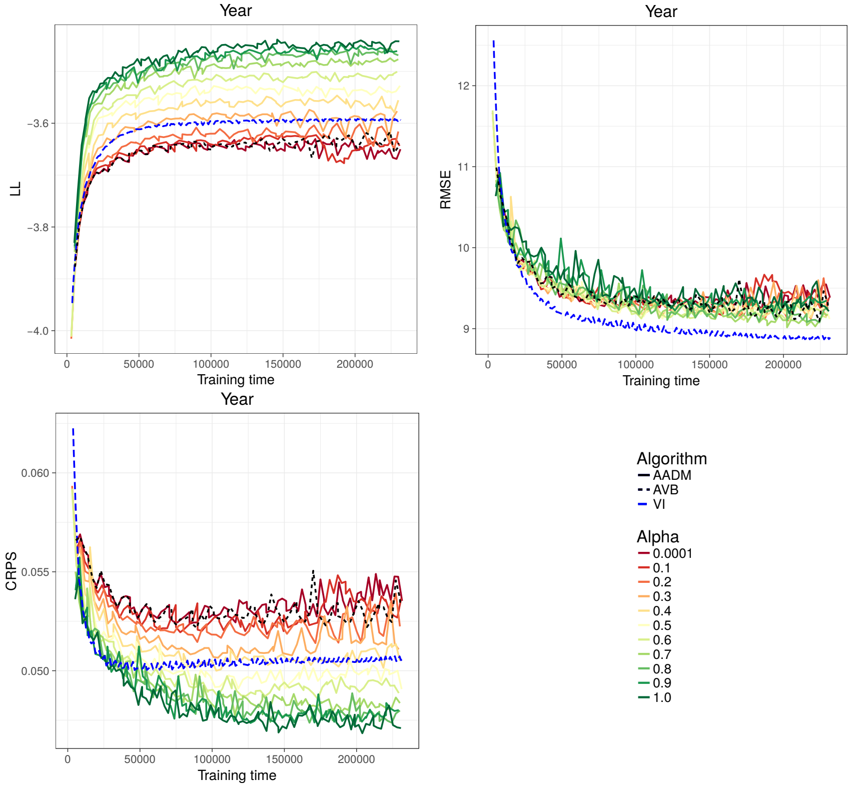

Figure 9: Performance as a function of the computational time in the Year dataset for each method. We report both in test log-likelihood (top-left), RMSE (top-right) and CRPS (bottom). The dashed blue line corresponds to the method VI, the dashed black line to AVB, and other solid lines represent our method, AADM, for different values of alpha. Best seen in color.

The results obtained for each method on the Year dataset are displayed in Figure 9. Again, in this figure dashed lines represent other methods, the black being AVB and the blue VI. Solid lines represent our method, AADM, for different values of alpha. As in the previous dataset, AADM obtains better results than AVB and VI in terms of both the test log-likelihood and the CRPS when α approaches 1. When α is closer to 0, AADM, gives similar results to those of AVB and VI. Specially, in the case of the CRPS we see that the VI performs better than both AVB and AADM, for α between 0 and 0.5. However, when α increases, AADM outperforms all of the previous methods. In terms of

RMSE, lower values of α seems to give also the best results. However, in this case higher values of α do not seem to give significantly worse results in terms of this metric.

7. Conclusions

An estimate of the uncertainty in the predictions made by machine learning algorithms like neural networks is of paramount importance in some specific applications. This estimate can be obtained by following a Bayesian approach. More precisely, the posterior distribution captures which model parameters (neural network weights) are compatible with the observed data. The posterior distribution can then be used to compute a predictive distribution that summarizes the uncertainty in the predictions made. A difficulty, however, is that computing the posterior distribution is intractable and one has to resort to approximate methods in practice.

In this paper we have described a general method for approximate Bayesian inference. The method proposed, called Adversarial α-divergence Minimization (AADM), allows to tune an approximate posterior distribution by approximately minimizing the α-divergence between this distribution and the posterior. The α-divergence generalizes the KL divergence, commonly used to perform approximate inference. AADM also allows to account for implicit models in the approximate posterior distribution. Implicit models allow to specify a probability distribution simply as some non-linear transformation of random input noise. If the non-linear transformation is complex enough, this will lead to a flexible model that is able to represent arbitrarily complex posterior distributions. A drawback of implicit models is, however, that one cannot evaluate the p.d.f. of the resulting distribution, which is required for approximate inference. We overcome this problem by following the approach described in [10]. More precisely, we learn a discriminative model that estimates the log-ratio between the p.d.f. of the implicit model and a much simpler distribution (*i.e.*, a Gaussian distribution).

The proposed method has been evaluated on several experiments and compared to other methods for approximate inference such as Variational Inference (VI) with a factorizing Gaussian as the approximate distribution, and Adversarial Variational Bayes (AVB) [10]. The experiments carried out, involving approximate inference with Bayesian neural networks, indicate that implicit models almost always provide better results than a factorizing Gaussian in terms of the metrics employed. Moreover, in regression tasks, the minimization of α-divergences seems to provide overall better results than the plain minimization of the KL divergence, as done by VI and AVB. In particular, values of α that are close, but not exactly equal to 1 seem to provide better predictive distributions in terms of the test log-likelihood and the CRPS metric. By contrast, in terms of the root mean squared error (RMSE) one should choose values of α that are close to, but not exactly equal to zero.

The approximate minimization of the α-divergence has been shown empirically to provide better results than the minimization of the KL divergence that is used in VI and AVB. More precisely, the proposed method, AADM, allows to capture patterns in the predictive distribution such as heteroscedastic noise or multiple modes. By contrast, these patterns are ignored when the typical KL divergence is minimized. This a consequence of using higher values for the α parameter that lead to a more inclusive behavior of the divergence. Specifically, higher values of α are expected to avoid that the approximate posterior distribution does not have high probability density in those regions of the parameter space in which the exact posterior has high probability density.

Therefore, we conclude that one can obtain better results in terms of the quality of the predictive distribution (such as the RMSE, the test log-likelihood, or the CRPS) by employing the proposed method, AADM, and by choosing a value of α that may depend on the specific performance metric we are interested in. A better predictive distribution can be obtained, in terms of the RMSE, the test log-likelihood or the CRPS, by using intermediate values of α. In general, however, there is no simple way of choosing an adequate value of α for each task. Our recommendation is that if one is interested in a small prediction error, one should use small values for α. By contrast, if one is interested in more accurate predictive distributions in terms of the test log-likelihood or the CRPS, larger values for α are preferred. Ideally, one should carry out a cross validation procedure to choose the optimal value for α.

Acknowledgements

Simón Rodríguez acknowledges the Spanish Ministry of Economy for the FPI SEV-2015-0554-16-4 Ph.D. grant. The authors gratefully acknowledge the use of the facilities of Centro de Computación Científica (CCC) at Universidad Autónoma de Madrid. Daniel Hernández-Lobato also acknowledges financial support from PID2019-106827GB-I00 / AEI / 10.13039/501100011033.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[3] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.

[4] Y. Gal, Uncertainty in deep learning, Ph.D. thesis, PhD thesis, University of Cambridge (2016).

[5] R. M. Neal, Bayesian learning for neural networks, Vol. 118, Springer Science & Business Media, 2012.

[6] J. M. Hernández-Lobato, R. P. Adams, Probabilistic backpropagation for scalable learning of Bayesian neural networks, in: International Conference on Machine Learning, 2015, pp. 1861–1869.

[7] J. M. Hernández-Lobato, Y. Li, M. Rowland, D. Hernández-Lobato, T. D. Bui, R. E. Turner, Black-box α-divergence minimization (2016) 1511–1520.

[8] A. Graves, Practical variational inference for neural networks, in: Advances in neural information processing systems, 2011, pp. 2348–2356.

[9] D. J. Rezende, S. Mohamed, Variational inference with normalizing flows, in: International Conference on Machine Learning, 2016, pp. 1530–1538.

[10] L. Mescheder, S. Nowozin, A. Geiger, Adversarial Variational Bayes: Unifying variational autoencoders and generative adversarial networks, in: International Conference on Machine Learning, 2017, pp. 2391–2400.

[11] Q. Liu, D. Wang, Stein variational gradient descent: A general purpose Bayesian inference algorithm, in: Advances In Neural Information Processing Systems, 2016, pp. 2378–2386.

[12] T. Salimans, D. Kingma, M. Welling, Markov chain Monte Carlo and variational inference: Bridging the gap, in: International Conference on Machine Learning, 2015, pp. 1218–1226.

[13] D. Tran, R. Ranganath, D. Blei, Hierarchical implicit models and likelihood-free variational inference, in: Advances in Neural Information Processing Systems, 2017, pp. 5523–5533.

[14] Y. Li, Q. Liu, Wild variational approximations, in: NIPS workshop on advances in approximate Bayesian inference, 2016.

[15] T. Gneiting, A. E. Raftery, Strictly proper scoring rules, prediction, and estimation, Journal of the American statistical Association 102 (2007) 359–378.

[16] M. J. Beal, Variational algorithms for approximate Bayesian inference, Ph.D. thesis (2003).

[17] C. Blundell, J. Cornebise, K. Kavukcuoglu, D. Wierstra, Weight uncertainty in neural network, in: International Conference on Machine Learning, 2015, pp. 1613–1622.

[18] F. Huszár, Variational inference using implicit distributions, ArXiv preprint arXiv:1702.08235 (2017).

[19] S. Amari, Differential-geometrical methods in statistics, Vol. 28, Springer Science & Business Media, 2012.

[20] T. Minka, Divergence measures and message passing, Tech. rep., Technical report, Microsoft Research (2005).

[21] B. J. Frey, R. Patrascu, T. Jaakkola, J. Moran, Sequentially fitting ”inclusive” trees for inference in noisy-OR networks, in: Advances in Neural Information Processing Systems, 2001, pp. 493–499.

[22] T. P. Minka, Expectation propagation for approximate bayesian inference, in: Uncertainty in Artificial Intelligence, 2001, pp. 362–369.

[23] Y. Li, Y. Gal, Dropout inference in bayesian neural networks with alpha-divergences, in: International Conference on Machine Learning, 2017, pp. 2052–2061.

[24] T. P. Minka, Power EP, Tech. rep., Technical report, Microsoft Research, Cambridge (2004).

[25] T. Heskes, O. Zoeter, Expectation propagation for approximate inference in dynamic Bayesian networks, in: Uncertainty in Artificial Intelligence, 2002, pp. 216–223.

[26] A. Rényi, On measures of entropy and information, in: Fourth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, 1961, pp. 547–561.

[27] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, O. Winther, Ladder variational autoencoders, in: Advances in neural information processing systems, 2016, pp. 3738–3746.

[28] S. Duane, A. D. Kennedy, B. J. Pendleton, D. Roweth, Hybrid Monte Carlo, Physics letters B 195 (1987) 216–222.

[29] R. M. Neal, MCMC using Hamiltonian dynamics, in: Handbook of Markov chain Monte Carlo, 2011, pp. 113–162.

[30] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, L. K. Saul, An introduction to variational methods for graphical models, Machine learning 37 (1999) 183–233.

[31] D. Soudry, I. Hubara, R. Meir, Expectation backpropagation: Parameter-free training of multilayer neural networks with continuous or discrete weights, in: Advances in Neural Information Processing Systems, 2014, pp. 963–971.

[32] M. D. Hoffman, Learning deep latent Gaussian models with Markov chain Monte Carlo, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017, pp. 1510–1519.

[33] M. K. Titsias, F. J. R. Ruiz, Unbiased implicit variational inference, in: Artificial Intelligence and Statistics, 2019, pp. 167–176.

[34] Y. Gal, Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, in: International Conference on Machine Learning, 2016, pp. 1050–1059.

[35] D. Dua, C. Graff, UCI machine learning repository (2017).
URL `http://archive.ics.uci.edu/ml`

[36] D. P. Kingma, J. Ba, ADAM: A method for stochastic optimization, in: International Conference on Learning Representations, 2015.

[37] E. P. Grimit, T. Gneiting, V. J. Berrocal, N. A. Johnson, The continuous ranked probability score for circular variables and its application to mesoscale forecast ensemble verification, Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography 132 (621C) (2006) 2925–2942.

[38] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, S. Udluft, Learning and policy search in stochastic dynamical systems with Bayesian neural networks, arXiv preprint arXiv:1605.07127.

[39] J. Hensman, N. Fusi, N. D. Lawrence, Gaussian processes for big data, in: Uncertainty in Artificial Intellegence, 2013, pp. 282–290.