Predictive Uncertainty in Large Scale Classification using Dropout - Stochastic Gradient Hamiltonian Monte Carlo.

Vergara, Diego\textsuperscript{1}, Hernández, Sergio\textsuperscript{2}, Valdenegro-Toro, Matías\textsuperscript{3} and Jorquera, Felipe\textsuperscript{4}.
\textsuperscript{1}Laboratorio de Procesamiento de Información Geoespacial, Universidad Católica del Maule, Chile.
\textsuperscript{2}German Research Centre for Artificial Intelligence, Bremen, Germany.
Email: \textsuperscript{1}diego.vergara@alu.ucm.cl, \textsuperscript{2}shernandez@ucm.cl, \textsuperscript{3}matias.valdenegro@dfki.de, \textsuperscript{4}f.jorquera.uribe@gmail.com

\textbf{Abstract}—Predictive uncertainty is crucial for many computer vision tasks, from image classification to autonomous driving systems. Hamiltonian Monte Carlo (HMC) is an inference method for sampling complex posterior distributions. On the other hand, Dropout regularization has been proposed as an approximate model averaging technique that tends to improve generalization in large scale models such as deep neural networks. Although, HMC provides convergence guarantees for most standard Bayesian models, it does not handle discrete parameters arising from Dropout regularization. In this paper, we present a robust methodology for predictive uncertainty in large scale classification problems, based on Dropout and Stochastic Gradient Hamiltonian Monte Carlo. Even though Dropout induces a non-smooth energy function with no such convergence guarantees, the resulting discretization of the Hamiltonian proves empirical success. The proposed method allows to effectively estimate the predictive accuracy and to provide better generalization for difficult test examples.

\textbf{Index Terms}—Uncertainty, Hamiltonian Monte Carlo, Dropout, Large Scale, High Dimensional, Classification.

I. INTRODUCTION

Artificial Intelligence systems have a wide variety of applications today, in some cases, are part of complex systems whose operation involves making delicate and dangerous decisions \cite{1}. Uncertainty in knowledge representation and reasoning has been studied since the fall of symbolic expert systems [2]. Therefore, several research efforts focused on efficient methods for estimating model uncertainty and capturing the variability inherent to real world situations [3].

Surprisingly, most modern artificial intelligence based computer vision techniques are generally unable to estimate their uncertainty. Predictive uncertainty is crucial for many tasks, such as image classification, detecting noisy examples (adversarial examples), to analyze failure cases in decision-making systems [4]. These problems depend on the uncertainty estimates for achieving good performance, requiring well calibrated probabilities. In such cases, Bayesian inference provides posterior predictive distributions that can be used to reduce over-confidence of the model outputs [5].

Hamiltonian Monte Carlo (HMC) is a Markov Chain Monte Carlo (MCMC) method for obtaining a sequence of random samples while maintaining asymptotic consistency with respect to a target distribution \cite{6}, \cite{7}. HMC provides a mechanism for defining proposals with high acceptance rate, enabling more efficient exploration of the state space than standard random-walk proposals. In addition, another property of HMC is the feasibility to support high dimensionality models by using the conservation of energy principle of dynamical systems [8]. This is largely due to the capability to adapt to the problem geometry using the gradient of the energy function. Moreover, the method can be extended to support large-scale or streaming data sets [9].

MCMC techniques that account for model uncertainty may also use sparsity-promoting priors such as the spike and slab prior \cite{10}, Bayesian Lasso \cite{11} and the Horseshoe prior \cite{12}. However, because of the combinatorial nature of the proposal distributions, MCMC techniques scale poorly in high dimensional models and variational inference must be introduced \cite{13}. Dropout has been previously proposed as a regularization technique for deep neural networks \cite{14}. The relation with model uncertainty arises from the approximate averaging variational distribution \cite{15}. However, variational estimates can be inaccurate or overly concentrated in the posterior models. Therefore, in this work, we propose a methodology named Dropout - Stochastic Gradient Hamiltonian Monte Carlo (D-SGHMC). The proposed approach is tested with the MNIST digit recognition \cite{16} and the Adience age recognition \cite{17} data sets. Predictive uncertainty estimates are compared to the estimates obtained by Stochastic Gradient HMC (SGHMC) \cite{9} and Stochastic Gradient Langevin Dynamics (SGLD) \cite{18}.

II. RELATED WORK

Many HMC variants based are found today in literature \cite{19}. HMC uses a discrete approximation of a continuous dynamic system. One such approach is the leapfrog integrator, which provides a numerical scheme to simulate Hamiltonian dynamics. Nevertheless, the leapfrog algorithm is highly sensitive to critical user-specified hyper-parameters, such as the number of steps and the constant step size. Hoffman \textit{et. al.} presents the No-U-Turn sampler (NUTS) \cite{20}, which is an extension of HMC that automatically determines the appropriate number of leapfrogs steps that the algorithm needs in order to converge to
the target distribution. This method uses a recursive algorithm to build a set of likely candidate points, stopping automatically when it starts to double back and retrace its steps. On the other hand, simple heuristics can also be used to establish the number of leapfrogs steps and the step-size, based on the assumption that the posterior distribution is Gaussian with diagonal covariance [21].

Other highly successful implementations focus on attacking large data sets using previous ideas from stochastic optimization such as Stochastic Gradient Descent (SGD). The SGHMC and SGLD techniques support data batches and introduces a novel integrator using Langevin dynamics which takes into account the extra-noise induced in the gradient [9], [18]. Another technique, the Metropolis Adjusted Langevin Algorithm (MALA) [22], [23], uses a Metropolis-Hastings (Metropolis) correction scheme to accept or reject proposals. Also, the Riemann Manifold Hamiltonian Monte Carlo method provides better adaptation to the problem geometry for strongly correlated densities (RMHMC) [24], [25]. Finally, proximal algorithms and convex optimization techniques have also been studied in the context of MCMC [26], and HMC with log-concave or non-differentiable (non-smooth) energy functions [27].

The computational cost has been also discussed, especially with the apogee of large data volumes and the emergence of Deep Neural Networks (DNNs). Split HMC [28] is a technique that accelerates the computation assuming that it is possible to divide the Hamiltonian function. On the other hand, solutions based on massively parallel computing and Graphics Processing Units (GPUs) have been also proposed [29].

In the context of computer vision, Stochastic Gradient - MCMC (SG-MCMC) has been proposed to model weight uncertainty for shape classification [30]. Li et al. propose an scalable learning approach for DNNs, reinforcing the connection from stochastic optimization and MCMC techniques. Moreover, it has been shown that SGD with Gaussian Drop-Connect as an alternative to Dropout, shares the same form as SGLD. Conversely, integrating Dropout with SG-MCMC can be seen as a model average of mixtures of neural networks.

III. METHODS

In this section some important concepts for understanding the proposed method are detailed.

A. Hamiltonian Monte Carlo

Given a random variable \( \theta \in \mathbb{R}^n \) and a data set \( D = (X, Y) \), we want to sample from the posterior distribution, such that:

\[
p(\theta | D) = \frac{p(D | \theta)p(\theta)}{p(D)} = \frac{p(D | \theta)p(\theta)}{\int p(D | \theta)p(\theta) d\theta} \quad (1)
\]

HMC [6] performs a physical simulation of a conservative system, where a Hamiltonian function \( H \) composed as a potential energy \( U \) and a kinetic energy \( K \). These terms are constructed as follows:

\[
H(\theta, r) = U(\theta) + K(r),
\]

where \( r \in \mathbb{R}^n \) is called the momentum and is considered as an auxiliary variable. A positive-definite mass matrix \( M \) is also introduced as follow:

\[
U(\theta) = -\sum_{d \in D} \log p(d | \theta) - \log p(\theta),
\]

\[
K(r) = \frac{1}{2} r^T M^{-1} r.
\]

Now, in order to sample from \( p(\theta | D) \), the method simulate Hamiltonian dynamics while leaving the joint distribution \( \pi(\theta, r) \) invariant, such that:

\[
p(\theta | D) \propto \exp(-U(\theta)),
\]

\[
\pi(\theta, r) \propto \exp(-H(\theta, r)).
\]

The first step proposes a new value for the momentum from a Gaussian distribution. Then, a Metropolis update using Hamiltonian dynamics is used to propose a new state. The state evolves in a fictitious continuous time \( t \), and the partial derivatives of the Hamiltonian can be seen in Equation [5]

\[
d\theta = \frac{1}{2} M^{-1} r dt
\]

\[
dr = -\nabla U(\theta) dt
\]

In order to simulated continuous dynamics, the leapfrog integrator can be used to discretize time. Using a small step size \( \epsilon \), Equation [5] becomes:

\[
\theta^{(t + \frac{\epsilon}{2})} = \theta^{(t)} - \frac{\epsilon}{2} \nabla U(\theta^{(t)})
\]

\[
\theta^{(t + \epsilon)} = \theta^{(t + \frac{\epsilon}{2})} + \epsilon r^{(t + \frac{\epsilon}{2})} M^{-1}
\]

\[
r^{(t + \frac{\epsilon}{2})} = r^{(t)} + \frac{\epsilon}{2} \nabla U(\theta^{(t + \epsilon)})
\]

After \( i = 1, \ldots, m \) iterations of the leapfrog integrator with finite \( \epsilon \), the joint proposal becomes \( (\theta^{(t + 1)}), r^{(t + 1)} \) = \( (\theta_m, r_m) \). Subsequently, a Metropolis update is used to accept the proposal with probability \( \rho > \text{unif}(0, 1) \), such that:

\[
\rho = \min\{1, \exp(-H(\theta^{(t + 1)}, r^{(t + 1)})) + H(\theta^{(t)}, r^{(t)})\}
\]

In order to make a prediction \( Y^* \) at a new input \( X^* \), there is a need to calculate the predictive distribution:

\[
p(Y^* | X^*) = \int p(Y^* | X^*, \theta)p(\theta | D) d\theta
\]
1) Properties of Hamiltonian dynamics: An integrator is an algorithm that numerically approximates an evolution of the exact solution of a differential equation. Some important properties of Hamiltonian dynamics are important for building MCMC updates [7]:

- Reversibility: The dynamics are time-reversible.
- Volume Preservation: Hamiltonian dynamics are volume preserving.
- Conservation of the Hamiltonian: $H$ is constant as $\theta$ and $r$ vary.

The leapfrog method in HMC satisfies the criteria of volume conservation and reversibility over time. However, the total energy is only conserved approximately, in this way a bias is introduced in the joint density $(\theta, r)$. Conversely, the Metropolis update is used to satisfy the detailed balance condition.

2) Limitations of HMC: One of the main limitations of HMC is the lack of support for discrete parameters. The difficulty in extending HMC to a discrete parameter space stems from the fact that the construction of proposals relies on the numerical solution of a differential equation. In other hand, approximating the likelihood of a discrete parameter by a continuous density is not always possible [51]. Moreover, when any discontinuity is introduced into the energy function $(U(\theta))$, the first-order discretization does not ensure that a Metropolis correction maintains the stationary distribution invariant. Therefore, the standard implementation of HMC does not guarantee an acceptance rate when the parameter of interest has a discontinuous density. This is because integrators are designed for differential equations with smooth derivatives over time.

B. Stochastic Gradient Hamiltonian Monte Carlo

The standard HMC implementation needs to load all data in memory, which is not always possible when the volume of data increases considerably. Therefore, it is necessary to have an incremental integrator that satisfies the properties of Hamiltonian dynamics.

The SGHMC algorithm enables large-scale and online sampling enabling rapidly exploration of the posterior distribution. A naive way to achieve this is by applying a SGD modification to the HMC integrator and assess the impact of the noisy gradient (Eq. 2). The noise injected in the system by the stochastic gradient no longer leads to Hamiltonian dynamics that leaves the target distribution invariant. Although, it is possible to correct this problem using a Metropolis step, this procedure requires costly computation on the entire dataset and might lead to low acceptance rates.

$$\nabla \tilde{U}(\theta) \approx \nabla U(\theta) + \mathcal{N}(0, B)$$  \hspace{1cm} (9)

Chen et al. [9] propose an additional friction term $F$ to the momentum update and shows that using second-order Langevin dynamics counteract the effects of the injected Gaussian noise $\mathcal{N}(0, B)$ (see Alg. 1). On the other hand, the Metropolis step is no longer needed due to the stability of the second-order discretization [23]. This approach automatically corrects the error from the first-order approximation, making it unnecessary to correct this discrepancy and ensuring convergence to the stationary distribution [24].

A friction term $F = CM^{-1}r$ (where $M$ is the mass matrix and $C$ the friction constant) helps to decrease the energy $H(\theta, r)$, thus reducing the influence of the noise. The covariance matrix $B$ depends on the current state and the sample size. This procedure can be seen as a second-order Langevin dynamics:

$$d\theta = M^{-1}r \, dt$$
$$dr = -\nabla U(\theta) \, dt - CM^{-1}r \, dt + \mathcal{N}(0, 2(C - B) \, dt) + \mathcal{N}(0, 2B \, dt)$$  \hspace{1cm} (10)

which delivers a new integrator and update step. Algorithm 1 explains the resulting method.

**Algorithm 1 SGHMC**

**Require:** Starting position $\theta^1 \sim \mathcal{N}(0, 1)$.

for $t = 1, 2, \ldots$ do

$r^t \sim \mathcal{N}(0, M)$
$(\theta_0, r_0) = (\theta^t, r^t)$

$\textit{Simulate Dynamics (update step)}$

for $i = 1$ to $m$ do

$\theta_i \leftarrow \theta_{i-1} + \epsilon_t M^{-1}r_{i-1}$
$r_i \leftarrow r_{i-1} - \epsilon_t \nabla \tilde{U}(\theta_i) - \epsilon_t CM^{-1}r_{i-1} + \mathcal{N}(0, 2(C - B)\epsilon_t)$

end for

$(\theta^{t+1}, r^{t+1}) = (\theta_m, r_m), \text{ no Metropolis-Hastings step.}$

end for

return Fully constructed Markov Chain

The dynamics given by Equation 10 have been shown to preserve the invariance properties of the original Hamiltonian dynamics, even with noise present [9].

C. Dropout / DropConnect

Dropout and DropConnect arise in the context regularization of DNNs and provide a way to combine exponentially many different architectures [14], [32]. In the field of DNN, Dropout/DropConnect can be described as follows:

- Regular (binary) DropConnect adds noise to global network weights, by setting a randomly selected subset of weights to zero within each layer:

$$z_a = g((\Gamma \odot \theta)h); \Gamma \sim \text{Bernoulli}(1 - p)$$  \hspace{1cm} (11)

- Binary Dropout instead randomly selects local hidden units:

$$z_b = \Gamma \odot g(\theta h); \Gamma \sim \text{Bernoulli}(1 - p)$$  \hspace{1cm} (12)

where $h$ and $z_{a,b}$ are the input and output layers, and $p$ is the probability that the weight/layer is dropped.
The Dropout learning interpretation has been the subject of research in recent years, due to its great effectiveness in various types of Neural Networks. Thus, has been shown that Dropout is similar to bagging and related ensemble methods \[^{[53]}\]. Since all models are averaged in an efficient and fast approximation with weights scaling, Dropout can be seen as an approximation to the geometric mean in the space of all possible models, this approximation is less expensive than the arithmetic mean in bagging methods \[^{[34]}\].

IV. DROPOUT - STOCHASTIC GRADIENT HAMILTONIAN MONTE CARLO

Gradient descent optimization has been extensively used for training Neural Networks architectures. In the context of Bayesian models, SGLD combines noisy gradient updates with Langevin dynamics in order to generate proposals from data subsets. Chen et al. proposes SGD with momentum as an alternative discretization \[^{[9]}\]. Conversely, SGHMC use a decaying learning rate that in the limit, reduces the discretization error to zero. The learning rate schedule ensures efficient sampling and high acceptance rates. Equation \[^{[13]}\] shows the SGHMC update.

\[
\begin{align*}
\Delta \theta &= v \\
\Delta v &= -\alpha v - \eta \nabla U(\theta) + \mathcal{N}(0, 2(\alpha - \beta)\eta)
\end{align*}
\] (13)

where \(v\) denotes the alternative to the momentum variable in SGHMC, \(\alpha = C/2\) is composed of a friction constant \(C\) that can be chosen as a finite small number, a step size \(\epsilon\) representing the initial value of the learning rate \(\eta\) and a noise model \(\beta = \eta B/2\). The momentum variable in SGD can be expressed as \(1 - \alpha\).

Incorporating Dropout can be seen as a regularization term that is directly related with the input data. On each iteration, a multivariate Bernoulli mask \(\Gamma = B(1 - p)\) with probability \(p\) is applied. Each one of the input components is randomly deleted and the remaining elements are scaled up by \(1/p\).

The noisy gradient updates generate proposals from a perturbed target distribution \[^{[35]}\]. The proposed method is described in algorithm \[^{[2]}\].

Algorithm 2: D-SGHMC

**Require:** Starting state \(\theta^1 \sim \mathcal{N}(0, 1)\).

**for** \(t = 1, 2, \ldots\) **do**

\(v^t \sim \mathcal{N}(0, M)\)

\((\theta_0, v_0) = (\theta^t, v^t)\)

**Simulate Dynamics (update step)**

\(\Gamma \sim B(1 - p)\)

\(X' \leftarrow X \odot (\Gamma \cdot 1/p)\)

**for** \(i = 1 \text{ to } m\) **do**

\(\theta_i \leftarrow \theta_{i-1} + v_{i-1}\)

\(v_i \leftarrow v_{i-1} - \alpha v_{i-1} - \eta \nabla U(\theta_i) + \mathcal{N}(0, 2(\alpha - \beta)\eta)\)

**end for**

\((\theta^{t+1}, v^{t+1}) = (\theta_m, v_m), \text{ no Metropolis-Hastings step.}\)

**end for**

**return** Fully constructed Markov Chain

For SGHMC introducing a second-order term reduces the discrepancy between the extra noise and the stationary distribution, so it is no longer necessary to make a Metropolis correction.

When Dropout is introduced, the energy function is no longer differentiable. In this context, the energy function of HMC requires specially tailored discretization. Nishimura \[^{[31]}\] finds a solution that preserves the critical properties of the Hamiltonian dynamics, through soft approximations, where the dynamics can be analytically integrated near the discontinuity in a way that preserves the total energy. \[^{[36]}\] have also shown that it is possible to build a discretization where the integrator maintains the irreversibility of the Markov chain and preserves energy, but volume preservation is no longer guaranteed \[^{[37]}\].

Dropout incorporates a discontinuous gradient to SGHMC, however, it is possible to evaluate the gradient of the method with automatic differentiation \[^{[38]}\], where the discontinuous parameters are taken to a continuous space \[^{[39]}\].

V. EXPERIMENTS

We implemented SGHMC, SGLD and D-SGHMC in Edward \[^{[40]}\], a Python library for posterior probabilistic modeling, inference, and criticism build on TensorFlow\[^{[1]}\]. We perform inference in two highly cited computer vision problems. These data sets were not selected with the objective of improving state of the art results, but to evaluate when we can incorporate uncertainty in the decision making on difficult examples.

In order to make an objective comparison between the studied methods and the proposed method, the optimal hyperparameters settings has been previously established. These settings are shown in Table \[^{[1]}\].

| Method | SGHMC | D-SGHMC | SGLD |
|--------|-------|---------|------|
| Step Size (\(\epsilon\)) | 0.0001 | - | - |
| Friction Constant (\(C\)) | 1.0 | - | - |
| Mini-Batch Size | 100 | - | - |
| Epochs | 100 | - | - |
| Warmup | 500 | - | - |
| Iterations | Epochs \(\times\) Num. Batches + Warmup | - | - |
| Prediction Samples | 30 | - | - |

Table I: Hyper-Parameters Setting

Training data is split in mini-batches of 100 in size and whitening is performed on each one of these batches. On the other hand, predictive distribution is estimated using 30 Monte Carlo samples over the Test dataset.

A. Digit Recognition on MNIST

In the first experiment, we decided to work with the MNIST \[^{[16]}\] dataset. This classic set of handwritten digits has served as the basis for the comparative evaluation of various classification algorithms. This database contains 60,000 training images and 10,000 test images, normalized to \(28 \times 28\) pixels

http://edwardlib.org

https://www.tensorflow.org
(784 features) and stored in gray scale. Figure 1 shows some examples of the database.

![Examples from the MNIST database.](image)

Fig. 1: Examples from the MNIST database.

A total number of 5 independent chains are performed for each method on MNIST. The Dropout probability is varied in $p \in [0.1, 0.5, 0.9]$, and the predictive results are compared with SGHMC and SGLD using the hyper-parameter setting established in the Table I. The comparison results can be observed in Table II.

| Method         | Total Accuracy (%) |
|----------------|--------------------|
| SGHMC          | 90.94 ± 0.28       |
| SGLD           | 88.06 ± 0.38       |
| D-SGHMC ($p = 0.9$) | 91.66 ± 0.10       |
| D-SGHMC ($p = 0.5$) | 91.72 ± 0.11       |
| D-SGHMC ($p = 0.1$) | 88.26 ± 0.08       |

TABLE II: Test Accuracy for MNIST data Set. 5 independent chains.

Compared with the other techniques, the model inference results show that D-SGHMC obtains a lower error when the Dropout probability is 0.5, obtaining state-of-the-art results in terms of linear classification methods [16].

The role of the Dropout probability ($p$) for D-SGHMC is also studied. Figure 2 shows accuracy results with the execution of 5 independent chains. It is possible to establish that for the studied data set a Dropout probability between on 0.6 and 0.8 generates lower error rates. On the other hand, also it can be seen that probabilities lower than 0.4 rapidly decreases the variable’s influence on the classification results.

Figure 3 shows the correlation between the true class and the predicted class. Higher class uncertainty is achieved when classifying digits 9 and 8 of MNIST, which could be classified as 4 and 5 respectively.

![Matrix Correlation for error rates on MNIST.](image-1)

Fig. 3: Matrix Correlation for error rates on MNIST.

Predictive accuracy is now evaluated by comparing the different methods, where the expected predictive accuracy is computed using a Monte Carlo approximation. D-SGHMC (Fig. 3a and 3b) achieves higher predictive accuracy when compared to SGHMC (Fig. 3c) and SGLD (Fig. 3d).

1) Confusing classes: One of the biggest challenges of the MNIST data set is to separate highly confusing digits such as ‘4’ and ‘9’. This problem has been addressed earlier in feature selection challenges [41].

We analyze the behavior between total uncertainty and predictive accuracy for one of these digits in each method.
We can observe that there is less over-confidence and at the same time the classification results improve. Figure 5 shows the mean expected probabilities for digit 9 in the test set.

As the Dropout rate decreases, the uncertainty around similar classes becomes more visible. However, low Dropout probabilities does not guarantee better classification results or improved uncertainty estimates. As shown in the Figure 5 models generated with low Dropout rates aggressively reduce the number of variables used for prediction.

2) Confusing Example (digit 9): In the first example (Fig. 6(a), both state-of-the-art SGHMC (Fig. 6(b) and SGLD methods (Fig. 6(c) maintain a high confidence.

In the example we can see how the proposed method decreases the over-confidence generated by SGHMC and SGLD and allows to classify the example effectively, producing higher uncertainty (Fig. 6(d), 6(e) and 6(f)).

3) Confusing Example (digit 8): In the second example (Fig. 7(a), we observe that it is similar to the digits ‘4’, ‘5’ and even ‘1’, for thus SGHMC (Fig. 7(b) and SGLD (Fig. 7(c) maintain a high confidence in this digits.

In this example, D-SGHMC does not significantly improve the classification results (Fig. 7(d) and 7(f)). However, it does improve the uncertainty estimate, generating higher confidence in the true label (Fig. 7(e)).

B. Age Recognition on ADIENCE

As part of the many challenges of facial recognition systems, age recognition has been recognized as difficult problem [17]. The ADIENCE dataset (see Fig. 8) contains 26,580 images of 2,284 different subjects and has been used to study the performance of age and gender recognition systems. After pre-processing and cleaning, the resulting data set is partitioned into 13,000 training examples and 3,534 test examples. The final number of samples can be seen in Table III.

Transfer learning from VGG-Face CNN (Convolutional Neural Networks) [42] with AVG pooling is used as a convolutional descriptor. For each example of ADIENCE database, VGG-Face computes a descriptor of 512 features.

A total number of 5 independent runs is used for each one of the methods. Using the hyper-parameter setting set in table I where state-of-art results are achieved [43]. Comparison with baseline methods can be seen in Table IV.
The role of the Dropout rate $p$ on the performance is analyzed. As shown in Figure 9 best results are obtained with values between 0.3 and 0.5.

In ADIENCE, uncertainty is increased for neighboring classes. This can be seen in Figure 10 which shows the true and the predicted labels.

### Table III: Final examples distribution of the ADIENCE data set.

| Age       | ID Class | Train | Test  | Total  |
|-----------|----------|-------|-------|--------|
| [0 – 2]   | 0        | 1.201 | 199   | 1.400  |
| [4 – 6]   | 1        | 1.566 | 573   | 2.139  |
| [8 – 13]  | 2        | 1.942 | 343   | 2.285  |
| [15 – 20] | 3        | 1.385 | 255   | 1.640  |
| [25 – 32] | 4        | 3.940 | 1.099 | 5.039  |
| [38 – 43] | 5        | 1.794 | 546   | 2.340  |
| [48 – 53] | 6        | 579   | 246   | 825    |
| [60 – ]   | 7        | 593   | 273   | 866    |
| **Total** |          | 8     | 13.990| 3.534  | 16.534 |

### Table IV: Test accuracy for ADIENCE data set.

| Method         | Total Accuracy (%) |
|----------------|--------------------|
| SGHMC          | 45.6 ± 1.23        |
| SGLD           | 44.0 ± 1.10        |
| D-SGHMC ($p = 0.9$) | 48.2 ± 0.56    |
| D-SGHMC ($p = 0.5$) | **51.6 ± 0.51**   |
| D-SGHMC ($p = 0.1$) | 48.3 ± 0.88      |

1) **Accuracy:** Class imbalance plays an important role in the final classification results for the age recognition problem in ADIENCE. D-SGHMC (see Figure 11d) improves results in most of classes, sacrificing to a lesser extent the accuracy for classes with less number of examples.

As the Dropout rate decreases, the imbalance problem becomes more evident and the accuracy results are also...
dropped (see Figure 9).

2) Predictive Distribution: In the ADIENCE data set there are many confusing examples which are usually neighboring classes. One example for each class is randomly sampled and the predictive distributions for each model are evaluated. These results are shown in Figure 12. In general, SGHMC and SGLD produce over-confident probabilities, assigning high values to incorrect labels. In contrast, D-SGHMC reduces that confidence, increasing uncertainty in the class predictions.

In particular, high confidence estimates for both SGHMC and SGLD on one example of class 4 can be seen in Figure 12. On the other hand, the proposed method allows to improve the uncertainty estimates between the neighboring classes, which significantly improves the classification results.

This situation is also replicated on one example of class 7, where SGHMC and SGLD achieve over-confident misclassification (class 1). The proposed method allows to alleviate the misclassification error and return uncertainty estimates for neighboring classes. The predictive distribution improves the classification results and provides the improved uncertainty estimates.
| Class   | SGHMC | SGLD | D-SGHMC ($p = 0.9$) | D-SGHMC ($p = 0.5$) | D-SGHMC ($p = 0.1$) |
|---------|-------|------|---------------------|---------------------|---------------------|
| Class 0 | ![Graph](image1) | ![Graph](image2) | ![Graph](image3) | ![Graph](image4) | ![Graph](image5) |
| Class 1 | ![Graph](image6) | ![Graph](image7) | ![Graph](image8) | ![Graph](image9) | ![Graph](image10) |
| Class 2 | ![Graph](image11) | ![Graph](image12) | ![Graph](image13) | ![Graph](image14) | ![Graph](image15) |
| Class 3 | ![Graph](image16) | ![Graph](image17) | ![Graph](image18) | ![Graph](image19) | ![Graph](image20) |
| Class 4 | ![Graph](image21) | ![Graph](image22) | ![Graph](image23) | ![Graph](image24) | ![Graph](image25) |
| Class 5 | ![Graph](image26) | ![Graph](image27) | ![Graph](image28) | ![Graph](image29) | ![Graph](image30) |
| Class 6 | ![Graph](image31) | ![Graph](image32) | ![Graph](image33) | ![Graph](image34) | ![Graph](image35) |
| Class 7 | ![Graph](image36) | ![Graph](image37) | ![Graph](image38) | ![Graph](image39) | ![Graph](image40) |

Fig. 12: Predictive distribution for randomly selected samples from the ADIENCE dataset.
VI. CONCLUSION

In many decision-making systems, the estimated uncertainty plays a crucial role in the final classification. Over-confident estimates can arise from noisy examples or when leading with classes that were not part of the training data set. The fundamental objective of achieving improved uncertainty estimates is to encourage decision-makers to question their decisions. In this way, compared to other state-of-the-art MCMC methods for large scale and high dimensional classification problems, the methodology presented in this paper improves the uncertainty estimated.

The proposed method is based on HMC and Dropout regularization. The experiments demonstrated that the method is capable of generating an approximation to the posterior distribution. In addition, the resulting predictive distributions also alleviate the misclassification error in difficult examples. However, it has not yet been proven that generated stochastic dynamics preserve the volume in its entirety. Future work will perform comparisons with other state-of-the-art variational methods. Moreover, the relationship between the proposed approach to approximate Bayesian model averaging can be also another line of research.

VII. ACKNOWLEDGMENTS

This work was supported by two CONICYT/FONDECYT grants, project Robust Multi-Target Tracking using Discrete Visual Features, code 11140598 and CONICYT/PFCHA Magister Nacional 2017 Scholarship, code 22170718.

REFERENCES

[1] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl, “Leveraging uncertainty information from deep neural networks for disease detection,” Scientific reports, vol. 7, no. 1, p. 17816, 2017.
[2] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.
[3] M. B. Christopher, Pattern Recognition and Machine Learning. Springer-Verlag New York, 2016.
[4] S. J. Prince, Computer vision: models, learning, and inference. Cambridge University Press, 2012.
[5] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, Bayesian data analysis. CRC press Boca Raton, FL, 2014.
[6] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, “Hybrid Monte Carlo,” Physics letters B, vol. 195, no. 2, pp. 216–222, 1987.
[7] R. M. Neal et al., “Mcmc using hamiltonian dynamics,” Handbook of Markov Chain Monte Carlo, vol. 2, no. 11, 2011.
[8] A. Beskos, N. Pillai, G. Roberts, J.-M. Sanz-Serna, A. Stuart et al., “Optimal tuning of the hybrid monte carlo algorithm,” Bernoulli, vol. 19, no. 5A, pp. 1501–1534, 2013.
[9] T. Chen, E. Fox, and C. Guestrin, “Stochastic gradient hamiltonian monte carlo,” in International Conference on Machine Learning, 2014, pp. 1683–1691.
[10] T. J. Mitchell and J. J. Beauchamp, “Bayesian variable selection in linear regression,” Journal of the American Statistical Association, vol. 83, no. 404, pp. 1023–1032, 1988.
[11] T. Park and G. Casella, “The bayesian lasso,” Journal of the American Statistical Association, vol. 103, no. 482, pp. 681–686, 2008.
[12] C. M. Carvalho, N. G. Polson, and J. G. Scott, “The horseshoe estimator for sparse signals,” Biometrika, vol. 97, no. 2, pp. 465–480, 2010.
[13] P. Carbonetto, M. Stephens et al., “Scalable variational inference for bayesian variable selection in regression, and its accuracy in genetic association studies,” Bayesian analysis, vol. 7, no. 1, pp. 73–108, 2012.
[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, 2014.
[15] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in international conference on machine learning, 2016, pp. 1050–1059.
[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
[17] E. Eizinger, R. Enbar, and T. Hassner, “Age and gender estimation of unfiltered faces,” IEEE Transactions on Information Forensics and Security, vol. 9, no. 12, pp. 2170–2179, 2014.
[18] M. Welling and Y. W. Teh, “Bayesian learning via stochastic gradient langevin dynamics,” in Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 681–688.
[19] M. Betancourt, “A conceptual introduction to hamiltonian monte carlo,” arXiv preprint arXiv:1701.02434, 2017.
[20] M. D. Hoffman and A. Gelman, “The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo,” Journal of Machine Learning Research, vol. 15, no. 1, pp. 1593–1623, 2014.
[21] M. D. Hoffman, “Learning deep latent gaussian models with markov chain monte carlo,” in International Conference on Machine Learning, 2017, pp. 1510–1519.
[22] G. O. Roberts and J. S. Rosenthal, “Optimal scaling of discrete approximations to langevin diffusions,” Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 60, no. 1, pp. 255–268, 1998.
[23] G. O. Roberts and O. Stramer, “Langevin diffusions and metropolis-hastings algorithms,” Methodology and computing in applied probability, vol. 4, no. 4, pp. 337–357, 2002.
[24] M. Girolami and B. Calderhead, “Riemann manifold langevin and hamiltonian monte carlo methods,” Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 73, no. 2, pp. 123–214, 2011.
[25] Z. Wang, S. Mohamed, and N. Freitas, “Adaptive hamiltonian and riemann manifold monte carlo,” in International Conference on Machine Learning, 2013, pp. 1462–1470.
[26] M. Pereyra, “Proximal markov chain monte carlo algorithms,” Statistics and Computing, vol. 26, no. 4, pp. 745–760, 2016.
[27] L. Chaari, J.-Y. Tourneret, C. Chaux, and H. Batatia, “A hamiltonian monte carlo method for non-smooth energy sampling,” IEEE Transactions on Signal Processing, vol. 64, no. 21, pp. 5585–5594, 2016.
[28] B. Shahbaba, S. Lan, W. O. Johnson, and R. M. Neal, “Split hamiltonian monte carlo,” Statistics and Computing, vol. 24, no. 3, pp. 339–349, 2014.
[29] A. L. Beam, S. K. Ghosh, and J. Doyle, “Fast hamiltonian monte carlo using gpu computing,” Journal of Computational and Graphical Statistics, vol. 25, no. 2, pp. 536–548, 2016.
[30] C. Li, A. Stevens, C. Chen, Y. Pu, Z. Gan, and L. Carin, “Learning weight uncertainty with stochastic gradient mcmc for shape classification,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5666–5675.
[31] A. Nishimura, D. Dunson, and J. Lu, “Discontinuous hamiltonian monte carlo for models with discrete parameters and discontinuous likelihoods,” arXiv preprint arXiv:1705.08510, 2017.
[32] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus, “Regularization of neural networks using dropout,” in Proceedings of the 30th international conference on machine learning (ICML-13), 2013, pp. 1058–1066.
[33] D. Warde-Farley, I. J. Goodfellow, A. Courville, and Y. Bengio, “An empirical analysis of dropout in piecewise linear networks,” arXiv preprint arXiv:1312.6197, 2013.
[34] P. Baldi and P. J. Sadowski, “Understanding dropout,” in Advances in neural information processing systems, 2013, pp. 2814–2822.
[35] R. Bardenet, A. Doucet, and C. Holmes, “Towards scaling up markov chain monte carlo: an adaptive subsampling approach,” in International Conference on Machine Learning, 2014, pp. 405–413.
[36] A. Pakman and L. Paninski, “Auxiliary-variable exact hamiltonian monte carlo method for non-smooth energy sampling,” in Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 681–688.
[37] H. M. Affshar and J. Domke, “Reflection, refraction, and hamiltonian monte carlo,” in Advances in Neural Information Processing Systems, 2015, pp. 3007–3015.
