Training Quantised Neural Networks with STE Variants: the Additive Noise Annealing Algorithm

Matteo Spallanzani[1*]   Gian Paolo Leonardi[2]   Luca Benini[1,3]

[1]Departement Informationstechnologie und Elektrotechnik, ETH Zürich, Switzerland

[2]Dipartimento di Matematica, Università di Trento, Italy

[3]Dipartimento di Ingegneria Elettrica e dell'Informazione, Università di Bologna, Italy

Abstract

Training quantised neural networks (QNNs) is a non-differentiable optimisation problem since weights and features are output by piecewise constant functions. The standard solution is to apply the straight-through estimator (STE), using different functions during the inference and gradient computation steps. Several STE variants have been proposed in the literature aiming to maximise the task accuracy of the trained network. In this paper, we analyse STE variants and study their impact on QNN training. We first observe that most such variants can be modelled as stochastic regularisations of stair functions; although this intuitive interpretation is not new, our rigorous discussion generalises to further variants. Then, we analyse QNNs mixing different regularisations, finding that some suitably synchronised smoothing of each layer map is required to guarantee pointwise compositional convergence to the target discontinuous function. Based on these theoretical insights, we propose additive noise annealing (ANA), a new algorithm to train QNNs encompassing standard STE and its variants as special cases. When testing ANA on the CIFAR-10 image classification benchmark, we find that the major impact on task accuracy is not due to the qualitative shape of the regularisations but to the proper synchronisation of the different STE variants used in a network, in accordance with the theoretical results.

1. Introduction

Deep learning has rapidly advanced during the last decade over a wide range of domains, from computer vision to natural language processing [3, 8]. However, making it pervasive requires deploying deep neural networks (DNNs) on embedded or edge devices, i.e., on computing systems with limited storage, memory, and processing capabilities. These constraints are at odds with the typical requirements of DNNs, which need millions or even billions of parameters and operations to deliver their performance.

Research in tiny machine learning (TinyML) has made considerable steps towards enabling the deployment of DNNs on resource-constrained devices. A first class of techniques aims at making DNNs more efficient in terms of accuracy-per-parameter or accuracy-per-operation [22, 23]. We refer to these techniques as *topological optimisations* since they revolve around achieving higher model efficiency by changing the structure of DNNs. A second class of approaches has instead focussed on deriving models that leverage the properties of the target deployment platform. These *hardware-related optimisations* include hardware-friendly activation functions [18], weight clustering and weight tensor decomposition [7, 13, 28], and QNNs [10].

QNNs use reduced-precision integer operands to meet the storage requirements and exploit the optimised support for integer arithmetic of embedded and edge platforms. With respect to their floating-point counterparts, QNNs typically introduce drops in task accuracy [11]. Several strategies have been proposed to counteract this shortcoming, ranging from changes to the target QNN's topology [16,30], through mixing different data representations and precisions inside the same network [17,24], to learning the shape of the stair functions used [4, 6, 12]. However, understanding how to propagate gradients through the discontinuous functions that model quantised operands remains a critical problem in QNN training.

Background & related work The classical derivative of a piecewise constant function is zero at all continuity points, while its distributional derivative is a linear combination of Dirac's deltas. Therefore, one can not directly apply the backpropagation algorithm to train a QNN. The standard solution to this problem is applying the so-called STE to all the discontinuous functions in the target QNN [2, 9]. Applying STE amounts to using two different functions during the forward and backward steps of the learning iteration,

---

*Corresponding author: spmatteo@ethz.ch

with the second one being differentiable. The choice of the replacement function is not unique. Previous research suggested that the alternative gradient computed using STE replacements is a descent direction for the so-called population loss, and that choosing proper backward functions is necessary to ensure convergence to a local minimum of the loss landscape [15, 27].

Several STE variants have been proposed in the literature aiming at maximising the task accuracy of the trained QNN [5, 17, 26]. It is worth noting that the problem of propagating gradients through non-differentiabilities is also relevant for spiking neural networks (SNNs) [20, 26]. Indeed, research in SNN training algorithms has proposed surrogate gradients similar to STE. In particular, the Whetstone method offers a solution to train an SNN from a DNN [20]: the method gradually transforms a DNN into an SNN by annealing the DNN's activation functions to the Heaviside step function during training. The annealing follows a heuristic schedule where the closer an activation is to the input, the sooner it is annealed. A similar soft-to-hard annealing has also been proposed to compress images into binary representations using autoencoders [1]. These dynamic STE variants add to the static variants proposed for QNNs. Is it possible to derive a unified description of static and dynamic STE variants? And what is their impact on QNN training?

Main contributions In this paper, we propose a theoretical framework to describe STE variants in a unified way. Specifically, we provide the following contributions to the field of QNN research:

- we observe that the backward functions associated with several STE variants proposed in the literature can be represented as the expected value of quantisers processing noisy inputs; this interpretation originates numerous families of STE variants;

- we analyse the problem of applying dynamic variants of STE to QNNs, introducing the novel concept of *compositional convergence*;

- we introduce ANA, a new algorithm to train QNNs encompassing standard STE and its variants (both static and dynamic ones) as special cases;

- when applying ANA to the CIFAR-10 image classification benchmark, we observe that the impact of the qualitative shape of the STE backward function on the final accuracy is at best minor; instead, we observe that the proper synchronisation of the regularisations in a QNN using dynamic STE variants is essential to guarantee convergence; the code to reproduce our experiments is available on GitHub[1].

The paper is organised as follows: in Section 2, we introduce the required terminology and prove the theoretical results; in Section 3, we describe ANA; in Section 4, we describe our experimental design and discuss its outcomes; finally, in Section 5, we summarise our findings.

2. Analysing STE variants

2.1. Quantisers and STE

Given an integer $K \geq 2$, a set $Q := \{q_0 < \cdots < q_{K-1}\} \subset \mathbb{R}$ of **quantisation levels** and a set $\Theta := \{\theta_1 < \cdots < \theta_{K-1}\} \subset \mathbb{R}$ of **thresholds**, we define a K**-quantiser** to be the stair function

$$
\begin{aligned}
\sigma_{\Theta,Q} \,:\, &\mathbb{R} \to Q \\
&x \mapsto q_0 + \sum_{k=1}^{K-1} (q_k - q_{k-1}) H_{\theta_k}^+(x) \,.
\end{aligned}
\tag{1}
$$

Here,

$$
\begin{aligned}
H_\theta^+ \,:\, &\mathbb{R} \to \{0, 1\} \\
&x \mapsto \begin{cases} 0, \text{if } x < \theta \,, \\ 1, \text{if } \theta \leq x \,, \end{cases}
\end{aligned}
\tag{2}
$$

is the (parametric) Heaviside function. Note that H_θ^+ is itself a 2-quantiser. For convenience, we define K-quantiser's **bins** to be the counterimages of the quantisation levels: $I_k := \sigma_{\Theta,Q}^{-1}(\{q_k\})$.

In practical applications, K is set to be equal to 2^B for some integer **precision** $B \geq 1$, and there exist an **offset** $z \in \mathbb{Z}$ and a **quantum** $\epsilon \in \mathbb{R}^+$ such that $q_0 = z\epsilon$ and $\theta_k = q_k = (z+k)\epsilon$ for $k = 1, \ldots, K-1$. This simplification allows rewriting (1) in terms of hardware-efficient flooring and clipping operations, yielding a **linear B-bit quantiser**: $\sigma_{\Theta,Q}(x) = \sigma_{z,\epsilon}(x) = \epsilon \operatorname{clip}(\lfloor x/\epsilon \rfloor, z, z + K - 1)$. z is usually chosen to be 0 (unsigned B-bit linear quantiser) or -2^{B-1} (signed B-bit linear quantiser).

When the exact value of K (respectively, B) is not relevant or can be inferred from the context, we will simply use the terms **quantiser** (respectively, **linear quantiser**). When clarity of exposition is not impacted, we will also drop the subscripts to ease readability.

Quantisers are piecewise constant functions: their classical derivative does not exist at the thresholds and is zero in the interior of the bins. This lack of differentiability is disruptive for backpropagation, and would theoretically prevent gradient-based training of QNNs.

The STE can be regarded as a technique to make a quantiser (1) differentiable by replacing it with a differentiable

---

[1] https://github.com/pulp-platform/quantlab/tree/ANA

or almost everywhere differentiable function $\tilde{\sigma}$ before computing the derivative. We name σ the **STE target** and $\tilde{\sigma}$ the **STE regularisation**. Example replacement functions for the Heaviside H_0^+ include the hard sigmoid $\tilde{\sigma}(x) := \max\{0, \min\{x + 1/2, 1\}\}$, the clipped ReLU $\tilde{\sigma}(x) := \max\{0, \min\{x, 1\}\}$, and the ReLU $\tilde{\sigma}(x) = \max\{0, x\}$.

2.2. Unifying STE variants

Consider a discontinuous function $\sigma : \mathbb{R} \to \mathbb{R}$. We say that a function $\sigma_\lambda : \mathbb{R} \to \mathbb{R}$ is a corresponding **parametric regularisation** if $\sigma_\lambda \in C^1(\mathbb{R})$ (or Lipschitz, and thus almost everywhere differentiable) for each $\lambda \in \mathbb{R}^+$, and $\lim_{\lambda \to 0} \sigma_\lambda(x) = \sigma(x), \forall x \in \mathbb{R}$; $\lambda \in \mathbb{R}^+$ is a parameter controlling the degree of regularisation.

It is elementary to show that the expectation operator acts as a convolution, transforming a discontinuous function into one that is differentiable, either in the classical or distributional sense. In what follows, we will overload μ to denote both an absolutely continuous probability measure on $\mathbb{R}$ and the corresponding probability density function.

Proposition 1. *Let $\sigma : \mathbb{R} \to Q$ be a K-quantiser. Let ν be a real random variable with probability density μ. For any value $\nu' \in \mathbb{R}$, we define the function $\sigma_{\nu'}(x) := \sigma(x - \nu')$. Then:*

(i) $\mathbb{E}_\mu[\sigma_\nu(x)] = (\mu * \sigma)(x), \ \forall\, x \in \mathbb{R}$; we therefore define $\mathbb{E}_\mu[\sigma_\nu] := \mu * \sigma$;

(ii) if $\mu \in W^{1,1}(\mathbb{R})$ then $\mathbb{E}_\mu[\sigma_\nu]$ is differentiable, its derivative is bounded, continuous, and satisfies $\frac{d}{dx}\mathbb{E}_\mu[\sigma_\nu] = D\mu * \sigma$;

(iii) if $\mu \in BV(\mathbb{R})$ then $D\mathbb{E}_\mu[\sigma_\nu] = D\mu * \sigma$ almost everywhere and it is bounded by $\|\sigma\|_\infty |D\mu|(\mathbb{R})$.

In other words, the proposition states that we can regularise a discontinuous function σ by convolving it with a probability density μ satisfying either $\mu \in W^{1,1}(\mathbb{R})$ (e.g., the triangular, normal, and logistic distributions) or the weaker $\mu \in BV(\mathbb{R})$ (e.g., the uniform distribution on a compact interval). Note that the noise density is an even function ($\mu(\nu) = \mu(-\nu)$) for common zero-mean distributions such as the uniform, triangular, normal, and logistic.

We can link the concept of regularised functions with the stochastic setting by considering a parametric density μ_λ. As an example, consider the uniform distribution whose mean $\alpha(\lambda)$ and standard deviation $\beta(\lambda)$ depend on λ in such a way that $\alpha(\lambda)$ and $\beta(\lambda)$ go to zero when $\lambda \to 0$. This distribution has density $\mu_\lambda(\nu) = \chi_{[a(\lambda),b(\lambda)]}(\nu)/(b(\lambda) - a(\lambda))$, where $a(\lambda) := \alpha(\lambda) - \sqrt{3}\beta(\lambda)$ and $b(\lambda) := \alpha(\lambda) + \sqrt{3}\beta(\lambda)$. Supposing that the target quantiser is $\sigma = H_0^+$,

we have

$$
\begin{aligned}
\sigma_\lambda(x) &:= (\mu_\lambda * \sigma)(x) \\
&= \begin{cases} 0, \text{ if } x < a(\lambda), \\ \frac{x - a(\lambda)}{b(\lambda) - a(\lambda)}, \text{ if } a(\lambda) \le x < b(\lambda), \\ 1, \text{ if } b(\lambda) \le x, \end{cases}
\end{aligned} \tag{3}
$$

which has derivative $(D\mu_\lambda * \sigma)(x) = \mu_\lambda(x)$. Since $\alpha(\lambda), \beta(\lambda) \to 0$ as $\lambda \to 0$, $\mu_\lambda \to \delta_0$ (the Dirac's delta centred at zero) in the distributional sense, and $\sigma_\lambda \to \sigma$ in the pointwise sense as requested by our definition of regularised function.

How does this discussion connect with STE? If we set $\alpha(\lambda) \equiv 1/2$ and $\beta(\lambda) \equiv 1/2\sqrt{3}$ in (3), σ_λ is the clipped ReLU. Similarly, if we set $\alpha(\lambda) \equiv 0$ and $\beta(\lambda) \equiv 1/2\sqrt{3}$, σ_λ is the hard sigmoid. The same principle can be adapted to derive the piecewise polynomial regularisation of H_0^+ proposed in [5, 17] (corresponding to a triangular noise distribution), as well as the error function (in case of normal noise) and the logistic function (in case of logistic noise). The observation that all these functions can be seen as regularisations of the Heaviside is not new [20, 26]; however, Proposition 1 generalises to broader classes of regularisations.

2.3. Dynamic STE and compositional convergence

In this sub-section, we consider the problem of regularising QNNs with STE variants that can evolve through time. First, we will set the formalism to describe arbitrary neural networks; after defining compositions of quantised layer maps, we will define compositions of regularised layer maps; finally, we will define the concept of compositional convergence and briefly discuss its implications

Let $L \ge 2$ be an integer **number of layers**. Given an integer **input size** $n_0 \ge 1$, let $X_0 \subset \mathbb{R}^{n_0}$ be the **input space**. For each $\ell = 1, \ldots, L$, define an integer **layer size** $n_\ell \ge 1$, a **feature space** $X_\ell \subseteq \mathbb{R}^{n_\ell}$, a space of **weight matrices** $W_\ell \subseteq \mathbb{R}^{n_\ell \times n_{\ell-1}}$, a space of **bias vectors** $B_\ell \subseteq \mathbb{R}^{n_\ell}$, and the **parameter space** $M_\ell := W_\ell \times B_\ell$.

For each $\ell = 1, \ldots, L$, given a fixed $\mathbf{m}_\ell = (\mathbf{W}_\ell, \mathbf{b}_\ell) \in M_\ell$, define the ℓ-th **layer map**

$$
\varphi_{\mathbf{m}_\ell} := \boldsymbol{\sigma}_\ell \circ S_{\mathbf{m}_\ell} \tag{4}
$$

as the composition of the affine map

$$
\begin{aligned}
S_{\mathbf{m}_\ell} &: X_{\ell-1} \to \mathbb{R}^{n_\ell} \\
\mathbf{x}_{\ell-1} &\mapsto \mathbf{W}_\ell \mathbf{x}_{\ell-1} + \mathbf{b}_\ell =: \mathbf{s}_\ell,
\end{aligned} \tag{5}
$$

and the element-wise map

$$
\begin{aligned}
\boldsymbol{\sigma}_\ell &: \mathbb{R}^{n_\ell} \to X_\ell \\
\mathbf{s}_\ell &\mapsto \boldsymbol{\sigma}_\ell(\mathbf{s}_\ell) = (\sigma_{\ell,1}(s_{\ell,1}), \ldots, \sigma_{\ell,n_\ell}(s_{\ell,n_\ell}))' =: \mathbf{x}_\ell,
\end{aligned} \tag{6}
$$

where the functions $\sigma_{\ell,i_\ell} : \mathbb{R} \to \mathbb{R}, i_\ell = 1, \ldots, n_\ell$, are **activation functions**. Activation functions are assumed to be non-constant and non-decreasing. For layer maps $\ell = 1, \ldots, L-1$, there must be at least one activation function σ_{ℓ,i_ℓ} which is non-linear (e.g., a bounded function). Both in theory and applications it is usually assumed that $\boldsymbol{\sigma}_L$ is the identity function.

Given $\bar{\ell} \in \{1, \ldots, L\}$, we define $\hat{\mathbf{m}}_{\bar{\ell}} := (\mathbf{m}_1, \ldots, \mathbf{m}_{\bar{\ell}})$ to be the collective parameter taken from $\widehat{M}_{\bar{\ell}} := M_1 \times \cdots \times M_{\bar{\ell}}$. We define a **network map** recursively as follows:

$$\begin{aligned} \Phi_{\hat{\mathbf{m}}_1} &:= \varphi_{\mathbf{m}_1}, \\ \Phi_{\hat{\mathbf{m}}_\ell} &:= \varphi_{\mathbf{m}_\ell} \circ \Phi_{\hat{\mathbf{m}}_{\ell-1}}, \ \ell = 2, \ldots, L. \end{aligned} \tag{7}$$

Consider a network map $\Phi_{\hat{\mathbf{m}}_L}$ such that all its activation functions $\sigma_{\ell,i_\ell}, i_\ell = 1, \ldots, n_\ell, \ell = 1, \ldots, L-1$ are the Heaviside (2). For each $\ell = 1, \ldots, L-1$, let $\sigma_{\lambda_\ell} : \mathbb{R} \to \mathbb{R}$ be a parametric regularisation of (2) with regularisation parameter $\lambda_\ell = \lambda_\ell(\lambda) > 0$, and $\boldsymbol{\sigma}_{\lambda_\ell} : \mathbb{R}^{n_\ell} \to \mathbb{R}^{n_\ell}$ be the component-wise application of σ_{λ_ℓ}. Analogously to (4), we define the ℓ-th **regularised layer map** as $\varphi_{\lambda_\ell,\mathbf{m}_\ell} := \boldsymbol{\sigma}_{\lambda_\ell} \circ S_{\mathbf{m}_\ell}$. Analogously to (7), we define the **regularised network map** as

$$\begin{aligned} \Phi_{\hat{\lambda}_1,\hat{\mathbf{m}}_1} &:= \varphi_{\lambda_1,\mathbf{m}_1}, \\ \Phi_{\hat{\lambda}_\ell,\hat{\mathbf{m}}_\ell} &:= \varphi_{\lambda_\ell,\mathbf{m}_\ell} \circ \Phi_{\hat{\lambda}_{\ell-1},\hat{\mathbf{m}}_{\ell-1}}, \ \ell = 2, \ldots, L. \end{aligned} \tag{8}$$

Given $\mathbf{x}_0 \in X_0$, we define

$$\mathbf{x}_\ell := \Phi_{\hat{\mathbf{m}}_\ell}(\mathbf{x}_0), \ \ell = 1, \ldots, L, \tag{9}$$

to be the **quantised features** of $\mathbf{x}_0$ and

$$\mathbf{x}_{\hat{\lambda}_\ell,\ell} := \Phi_{\hat{\lambda}_\ell,\hat{\mathbf{m}}_\ell}(\mathbf{x}_0), \ \ell = 1, \ldots, L. \tag{10}$$

to be the **regularised features** of $\mathbf{x}_0$. We are interested in understanding under which conditions on the evolution of σ_{λ_ℓ} the **regularised feature** $\mathbf{x}_{\hat{\lambda}_\ell,\ell}$ converges to the **quantised feature** $\mathbf{x}_\ell$. We recall that a **convergence rate** is a continuous and non-decreasing function $r : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ satisfying $\lim_{\lambda \to 0} r(\lambda) = 0$.

Theorem 2. *Consider a network map* (7) *parametrised by* $\hat{\mathbf{m}}^L = (\mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^L) \in \widehat{M}^L$ *and using the Heaviside* H_0^+ *as its activation function.*

Consider a regularised network map (8) *such that for* $\ell = 1, \ldots, L-1$ *the regularisations* σ_{λ_ℓ} *of* H_0^+ *and the regularisation parameters* λ_ℓ *satisfy the following conditions:*

$$\lambda_\ell \xrightarrow[\lambda \to 0]{} 0; \tag{11}$$

$$\sigma_{\lambda_\ell}(s) \xrightarrow[\lambda_\ell \to 0]{} H_0^+(s), \ \forall s \in \mathbb{R}; \tag{12}$$

$$\sigma_{\lambda_\ell} \text{ is strictly increasing}; \tag{13}$$

$$0 \le \sigma_{\lambda_\ell}(s) \le 1, \ \forall s \in \mathbb{R}. \tag{14}$$

Additionally, we assume that convergence rates $r_\ell(\lambda), \ell = 1, \ldots, L$ *are given, such that for every* $\varepsilon > 0$

$$\sigma_{\lambda_\ell}^{-1}(\varepsilon r_\ell(\lambda)) \xrightarrow[\lambda \to 0]{} 0, \tag{15}$$

$$\sigma_{\lambda_\ell}^{-1}(1 - \varepsilon r_\ell(\lambda)) \xrightarrow[\lambda \to 0]{} 0, \tag{16}$$

$$\frac{1 - \sigma_{\lambda_\ell}(0)}{r_\ell(\lambda)} \xrightarrow[\lambda \to 0]{} 0, \tag{17}$$

and, for $\ell = 2, \ldots, L,$

$$\frac{r_{\ell-1}(\lambda)}{\sigma_{\lambda_\ell}^{-1}(1 - \varepsilon r_\ell(\lambda))} \xrightarrow[\lambda \to 0]{} 0. \tag{18}$$

Then, for any given $\mathbf{x}^0 \in X^0$, *we have*

$$\frac{\|\mathbf{x}_{\hat{\lambda}_\ell,\ell} - \mathbf{x}_\ell\|}{r_\ell(\lambda)} \xrightarrow[\lambda \to 0]{} 0, \qquad \forall \ell = 1, \ldots, L. \tag{19}$$

If (11)-(18) hold, we say that the regularisations $\sigma_{\lambda_\ell}, \ell = 1, \ldots, L$ satisfy the **compositional convergence** hypothesis.

In other words, the theorem states that parametric regularisations should converge quantitatively faster for the layer maps closer to the input. Although this result describes how the regularised network processes information in the forward direction and can not explain the coherence between the regularised gradient (what is referred to as "coarse gradient" in [27]) and the population loss gradient, it provides a partial justification for the empirical findings of [20]. Indeed, the Whetstone method uses "annealing schedules" where the regularisations are stabilised in earlier layer maps before proceeding to later ones.

2.4. A connection between STE targets and STE regularisations

Consider an input x to a quantiser (1). When x is added noise following a distribution μ_λ, the probability of sampling the k-th quantisation level is $p_k = \mu_{\lambda,x}(\sigma^{-1}(\{q_k\})) = \mu_{\lambda,x}(I_k)$, where $\mu_{\lambda,x}(A) := \mu_\lambda(A - x), \forall A \in \mathcal{A}$ ($\mathcal{A}$ is the Borel σ-algebra on $\mathbb{R}$). When μ_λ is a zero-mean, uni-modal noise distribution, selecting the mode quantisation level $q_k' = \arg\max_{q_k \in Q} \mu_{\lambda,x}(I_k)$ returns the same value as applying the original σ directly to x. This observation is a more formal definition of what was referred to as "deterministic sampling" in [9]; we will make use of it in the next section.

3. The ANA algorithm

Consider the main propositions of Section 2:

- different STE regularisations can be modelled as the expected value of the corresponding STE target when it processes stochastic inputs;

- supposing that the noise distributions governing the parametric STE regularisations in a given feedforward network can evolve dynamically, we can ensure that the composition of the regularisations pointwise converges to the composition of the STE targets by enforcing the compositional convergence hypothesis;

- the STE target and STE regularisation can be seen as the same stochastic function operating according to two different strategies: mode in the forward pass, and expectation in the backward pass.

Based on these observations, we propose ANA, whose pseudo-code is listed in Algorithm 1. It takes in input an STE-regularised network map (8) initialised at $\hat{\mathbf{m}}_L^{(0)}$; a schedule $\mathcal{S}$ mapping tuples (ℓ, t) to the regularisation parameter λ_ℓ for the ℓ-th layer at the t-th iteration; a training data set $\mathcal{D} = ((x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)}))$, $N \geq 1$ being an integer number of data points; a loss function $\mathcal{L} : X_L \times X_L \to \mathbb{R}_0^+$; a learning rate $\eta \in \mathbb{R}^+$; an integer $T_e \geq 1$ number of training epochs. The routine set_noise performs look-ups from the schedule $\mathcal{S}$, whereas the routine optim computes the parameter updates. The total number of training iterations is $T := T_e N$. Of course, one can perform mini-batch stochastic gradient descent (SGD) (mini-batch size greater than one) instead of vanilla SGD.

Algorithm 1

Input:	$\Phi_{\hat{\lambda}_L, \hat{\mathbf{m}}_L^{(0)}}, \mathcal{S}, \mathcal{D}, \mathcal{L}, \eta, T_e$
Output:	$\Phi_{\hat{\lambda}_L, \hat{\mathbf{m}}_L^{(T)}}$
Uses:	set_noise, optim

1: $t \leftarrow 0$
2: **for** $t_e \leftarrow 1 : T_e$ **do**
3:     **for** (x, y') in $\mathcal{D}$ **do**
4:         $t \leftarrow t + 1$
5:         **for** $\ell \leftarrow 1 : L$ **do**
6:             $\lambda_\ell \leftarrow$ set_noise$(\mathcal{S}, \ell, t)$
7:         **end for**
8:         $y \leftarrow \Phi_{\hat{\lambda}_L, \hat{\mathbf{m}}_L^{(t-1)}}(x)$ ▷ inference
9:         $\mathbf{g}^{(t)} \leftarrow \nabla_{\hat{\mathbf{m}}_L} \mathcal{L}(y, y')$ ▷ backpropagation
10:        $\hat{\mathbf{m}}_L^{(t)} \leftarrow$ optim$(\hat{\mathbf{m}}_L^{(t-1)}, \mathbf{g}^{(t)}, \eta, t)$ ▷ update
11:     **end for**
12: **end for**
13: **return** $\Phi_{\hat{\lambda}_L, \hat{\mathbf{m}}_L^{(t)}}$

The first hyper-parameter of ANA is the collection $(\mu_{\lambda_1}, \ldots, \mu_{\lambda_L})$ of parametric probability measures used to regularise the layer maps $\varphi_{\lambda_\ell, \mathbf{m}_\ell}, \ell = 1, \ldots, L$.

The second hyper-parameter is the evolution of the regularisations as the training algorithm progresses. This hyper-parameter is encoded in the schedule $\mathcal{S}$ and controlled by Lines 5-7. Note that static schedules are also allowed as

special cases.

We also encoded a third hyper-parameter, which is not explicitly reported in Algorithm 1: the forward computation strategy. Given a layer map using a quantiser (1), we allow the forward pass to use either the expected value of the regularised quantiser, $\mathbb{E}_{\mu_{\lambda_\ell}}[\sigma_\nu]$, the mode $q_k' = \arg\max_{q_k \in Q}(p_k := \mu_{\lambda_\ell, x}(I_k))$, or a random sampling $q_k' \sim ((p_0, \ldots, p_{K-1}), 1)$. We name these strategies expectation, mode (or **deterministic sampling**) and random (or **random sampling**).

4. Experimental results

4.1. Methodology and experiment design

CIFAR-10 is a popular small data set for image classification [14]. It contains $60k$ RGB-encoded images partitioned into ten semantic classes. It consists of a training partition containing $5k$ images per class and a validation partition containing $1k$ images per class.

As a reference, we used a simple fully-feedforward network using five convolutional layers and three linear layers, inspired by the VGG family of topologies [21]; therefore, $L = 8$. We quantised all the weights and features to be ternary, apart from the last layer, which we kept in floating-point format coherently with common literature practice [4, 6].

In each experimental unit, we trained the network for 500 epochs using mini-batches of 256 images, the cross-entropy loss function, and the ADAM optimiser with an initial learning rate of 10^{-3}, decreased to 10^{-4} after 400 epochs.

Our experimental design consists of six degrees of freedom (DoFs): noise type, static mean, static variance, decay interval, decay power law, forward computation strategy. We detail their purpose in the following paragraphs.

The **noise type** is the parametric family of distributions used to instantiate $\mu_{\lambda_\ell}, \ell = 1, \ldots, L-1$. We considered four noise types: **uniform**, **triangular**, **normal** and **logistic**. We parametrised the distribution densities μ_{λ_ℓ} in their means $\alpha(\lambda_\ell)$ and standard deviations $\beta(\lambda_\ell)$:

$$\mu_{\lambda_\ell}(x) = \frac{\chi_{[\alpha(\lambda_\ell) - \sqrt{3}\beta(\lambda_\ell), \alpha(\lambda_\ell) + \sqrt{3}\beta(\lambda_\ell)]}(x)}{2\sqrt{3}\beta(\lambda_\ell)} \, ;$$

$$\mu_{\lambda_\ell}(x) = \begin{cases} 0, \text{if } x \notin [\alpha(\lambda_\ell) - \sqrt{6}\beta(\lambda_\ell), \alpha(\lambda_\ell) + \sqrt{6}\beta(\lambda_\ell)), \\ \frac{x - \alpha(\lambda_\ell) + \sqrt{6}\beta(\lambda_\ell)}{6\beta^2(\lambda_\ell)}, \text{if } x \in [\alpha(\lambda_\ell) - \sqrt{6}\beta(\lambda_\ell), \alpha(\lambda_\ell)), \\ \frac{\alpha(\lambda_\ell) + \sqrt{6}\beta(\lambda_\ell) - x}{6\beta^2(\lambda_\ell)}, \text{if } x \in [\alpha(\lambda_\ell), \alpha(\lambda_\ell) + \sqrt{6}\beta(\lambda_\ell)) \, ; \end{cases}$$

$$\mu_{\lambda_\ell}(x) = \frac{e^{-\frac{(x - \alpha(\lambda_\ell))^2}{2\beta^2(\lambda_\ell)}}}{\sqrt{2\pi}\beta(\lambda_\ell)} \, ;$$

$$\mu_{\lambda_\ell}(x) = \frac{e^{-\frac{x - \alpha(\lambda_\ell)}{\beta(\lambda_\ell)}}}{\beta(\lambda_\ell) \left(1 + e^{-\frac{x - \alpha(\lambda_\ell)}{\beta(\lambda_\ell)}}\right)^2} \, .$$

Note that all these distributions are uni-modal and have densities that are symmetric with respect to the mean.

Also, note that the uniform and triangular distributions' densities have compact support, whereas the normal and logistic have non-zero densities over all $\mathbb{R}$. To compare the regularisations obtained using compactly-supported and non-compactly-supported densities, we considered a non-compactly-supported measure μ_1 as equivalent to a compactly-supported measure μ_2 if they had the same mean and exactly 95% of the total probability mass of μ_1 fell in the support of μ_2.

The **noise schedule** defines how the measures $\mu_{\lambda_\ell}, \ell = 1, \ldots, L-1$ evolve through time. This evolution is controlled through the shape parameters $\alpha(\lambda_\ell), \beta(\lambda_\ell)$. In particular, denoting by $\alpha(\lambda_\ell^{(t)})$ and $\beta(\lambda_\ell^{(t)})$ the parameters of the distribution μ_{λ_ℓ} at the t-th iteration, their values are determined by the following functions:

$$\alpha(\lambda_\ell^{(t)}) = c_{\alpha,\ell} f_{\alpha,\ell}(\lambda_\ell^{(t)}),$$
$$\beta(\lambda_\ell^{(t)}) = c_{\beta,\ell} f_{\beta,\ell}(\lambda_\ell^{(t)}).$$

Here, $c_{\alpha,\ell}, c_{\beta,\ell}$ are non-negative real constants.

The **static mean** DoF is a Boolean variable. When it is set to true, then $f_{\alpha,\ell}(\lambda_\ell^{(t)}) \equiv 1$ and $c_{\alpha,\ell} = 0$. When it is set to false, then $c_{\alpha,\ell} > 0$ and

$$f_{\alpha,\ell}(\lambda_\ell^{(t)}) = (\lambda_\ell^{(t)})^{d_{\alpha,\ell}}$$
$$= \max\left\{0, \min\left\{\left(\frac{t_{\ell,end} - t}{t_{\ell,end} - t_{\ell,start}}\right), 1\right\}\right\}^{d_{\alpha,\ell}} \quad (20)$$

for some $0 \le t_{\ell,start} < t_{\ell,end} \le T$ and an integer $d_{\alpha,\ell} \ge 1$.

Similarly, the **static variance** DoF is a Boolean variable such that $f_{\beta,\ell}(\lambda_\ell^{(t)}) \equiv 1$ when it is set to true, and

$$f_{\beta,\ell}(\lambda_\ell^{(t)}) = (\lambda_\ell^{(t)})^{d_{\beta,\ell}}$$
$$= \max\left\{0, \min\left\{\left(\frac{t_{\ell,end} - t}{t_{\ell,end} - t_{\ell,start}}\right), 1\right\}\right\}^{d_{\beta,\ell}} \quad (21)$$

for some integer $d_{\beta,\ell} \ge 1$ when it is set to false. We always consider $c_{\beta,\ell} > 0$, since setting it to zero amounts to no regularisation and, therefore, zero gradients throughout the training process, resulting in the target network not being capable of learning.

When the static variance DoF is set to false, we assume that each distribution μ_{λ_ℓ} is annealed from some initial distribution $\mu_{\lambda_\ell^{(0)}}$ to the final distribution $\mu_{\lambda_\ell^{(T)}} = \delta_0$, which is a Dirac's delta centred at zero. For each distribution, we define two instants $0 \le t_{\ell,start} < t_{\ell,end} \le T$ such that $\mu_{\lambda_\ell^{(t)}} = \mu_{\lambda_\ell^{(0)}}, 0 \le t \le t_{\ell,start}$ and $\mu_{\lambda_\ell^{(t)}} = \mu_{\lambda_\ell^{(T)}}, t_{\ell,end} \le t \le T$. We call the sequence $\{t_{\ell,start}, \ldots, t_{\ell,end}\}$ the **annealing range** of μ_{λ_ℓ}.

The **decay interval** DoF is a categorical variable defining how the annealing ranges of the various measures μ_{λ_ℓ} are mutually related:

- **same start**: $t_{\ell,start}$ is the same for all layers, but $t_{\ell-1,end} < t_{\ell,end}$ for $\ell = 2, \ldots, L$;

- **same end**: $t_{\ell,start} < t_{\ell-1,start}$ for $\ell = 2, \ldots, L$, but $t_{\ell,end}$ is the same for all layers;

- **partition**: $t_{\ell-1,end} = t_{\ell,start}$ for $\ell = 2, \ldots, L$;

- **overlapped**: $t_{\ell_1,start} = t_{\ell_2,start}$ and $t_{\ell_1,end} = t_{\ell_2,end}$ for all $\ell_1, \ell_2 \in \{1, \ldots, L\}$.

The **decay power law** DoF is a binary categorical variable defining how the exponents $d_{\alpha,\ell}, d_{\beta,\ell}$ in (20) and (21) relate to each other. If it is set to **homogeneous**, then $d_{\alpha,\ell}$ and $d_{\beta,\ell}$ are the same for all the layers. If it is set to **progressive**, then $d_{\alpha,\ell}$ and $d_{\beta,\ell}$ are inversely proportional to the depth of the layer: this option enforces a faster annealing of μ_{λ_ℓ} to δ_0 over the corresponding decay range $\{t_{\ell,start}, \ldots, t_{\ell,end}\}$ in the layers that are closer to the input. In other words: the decay interval controls the relative precedence of annealing ranges, whereas the decay power law determines the speed of decay.

The last DoF of our experimental design is the **forward computation strategy**. As discussed in Section 3, it is a ternary categorical variable allowing the following options: **expectation**, **mode**, **random**.

The complete hyper-parameter search grid would consist of 384 configurations. However, we can avoid exploring a large subset of them by making some observations.

The combination of static mean and static variance determines the dynamics of the noise distributions; note that when both variables are set to true, then ANA is equivalent to standard STE. The combination of decay interval and decay power law defines the annealing schedule; when the distributions are static, that is, when we are in the case of standard STE, the value of this variable is not relevant, and we can fix it to an arbitrary value. Out of 96 experimental units that use static distributions, we need to explore only 12.

We did not consider the case where the static variance variable is set in combination with the expectation forward computation strategy. Indeed, the expected values of features and weights computed with respect to a non-collapsed noise distribution can greatly differ from the values computed with respect to a collapsed one; for this reason, the sudden removal of the noise at deployment time can disrupt the functional relationship learnt by the regularised network. This observation allows us to cut additional 32 out of 96 experimental units that use dynamic noise, static variance, and the expectation forward computation strategy, as well as 4 out of 12 units that use static noise distributions.
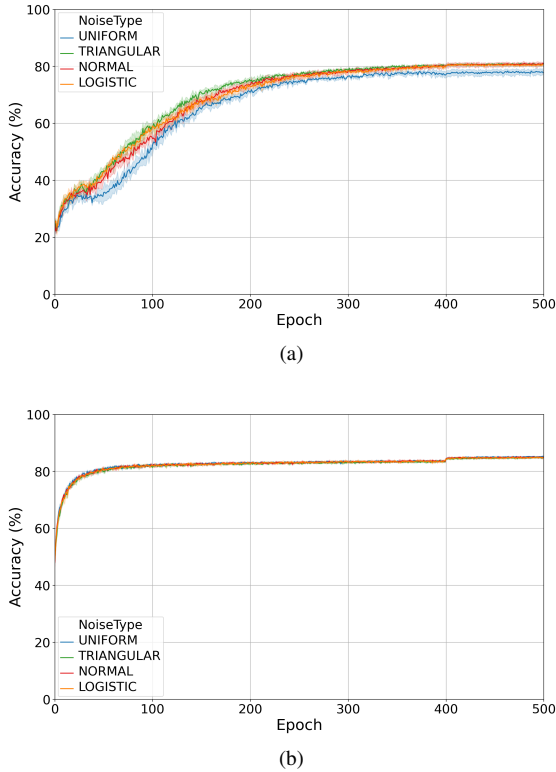
(a)

(b)

Figure 1. Performance of ANA using static noise schedules in combination with different forward computation strategies: random 1a, mode 1b. Different noise types are reported using different colours: uniform (blue), triangular (green), normal (red), logistic (yellow).

To assign confidence intervals to our measurements, we evaluated each hyper-parameter configuration using five-fold cross-validation on the CIFAR-10 training partition.

4.2. Results & discussion

Static noise schedules (i.e., schedules that do not anneal the distributions μ_{λ_ℓ} to δ_0) represent the baseline for our comparisons. Figures 1a, 1b show that there is a small advantage in using the mode forward computation strategy ($\sim 86\%$) as opposed to the random forward computation strategy ($\sim 82\%$). In part, this advantage is due to the higher sensitivity of the former strategy to the learning rate lowering that takes place at epoch 400.

In general, the noise type has negligible impact on task accuracy. There is an exception when using the random forward computation strategy: using the uniform noise yields slightly worse results than other distributions.

We start our analysis of **dynamic noise schedules** by considering the expectation forward computation strategy. As can be seen in Figure 2a, under a uniform noise distribution, the best convergence is achieved under the par-

tition and same start decay interval strategies ($\sim 79\%$ accuracy), with a slight advantage given when using homogeneous decay power laws as opposed to the progressive ones. However, the accuracy drop with respect to the static noise schedule baseline is not negligible (-7%). The same end decay interval strategy starts annealing the noise measure in the earlier layers only at a later moment during training, while the measures of the later layers are already converging, breaking the hypothesis of Theorem 2: indeed, we see that the corresponding configuration of ANA led to the worst performance over this batch of experimental units.

These observations were also confirmed for the triangular, normal, and logistic distributions, as shown in Figures 2b, 2c, 2d. As for the static noise schedule case, the noise type does not have a major impact on accuracy.

So far, experimental evidence suggested that the noise type is not the most relevant variable. Therefore, we fixed the noise type to uniform and analysed the impact of the forward computation strategy.

Figure 3a shows the performance of ANA under different schedules when random sampling is used in the inference pass. We can see that random sampling combined with the partition decay range strategy can improve accuracy by 4%. Although random sampling seems to mitigate the degradation due to the inappropriate scheduling associated with the same end decay interval strategy, this strategy remains the worst in this batch of experimental units.

Using deterministic sampling in combination with the partition decay interval strategy seems to allow filling the gap with the networks trained using static schedules, as shown in Figure 3b. We observe that the (small) advantage of static noise schedules is due to the learning rate lowering that takes place at epoch 400; in the chosen dynamic noise schedule, the regularising noise distributions are annealed by epoch 350, preventing gradients from tuning the parameters of all but the last layer. However, the two noise schedules yield the same accuracy until the learning rate lowering.

5. Conclusions

Propagating gradients through discontinuities remains a critical problem in training quantised neural networks. In this paper, we provided a unified framework to reason about STE and its variants. We formally showed how stochastic regularisation can be used to derive entire families of STE variants. Moreover, we analysed how dynamic STE variants can be used to regularise discontinuous networks and showed how to properly synchronise them to guarantee convergence to the target function during the inference pass. Our experiments on the CIFAR-10 benchmark highlighted that the major impact on accuracy is not due to the qualitative shape of the regularisers, but instead to the proper synchronisation of the STE variants used at different lay-
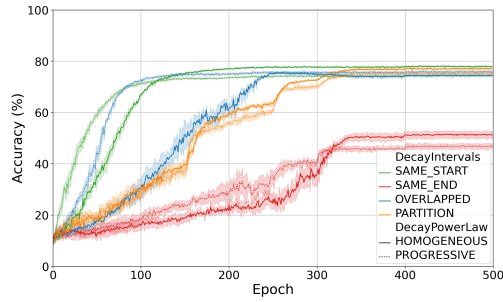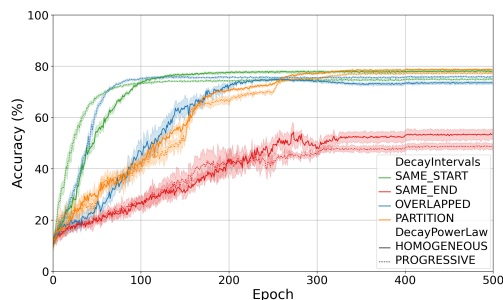
(a)
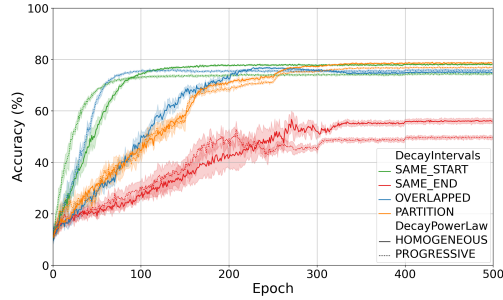
(b)
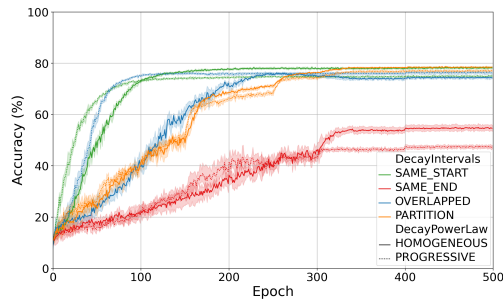
(c)

(d)

Figure 2. Performance of ANA using dynamic noise schedules under the expectation forward computation strategy and different noise types: uniform 2a, triangular 2b, normal 2c, logistic 2d. Each plot reports multiple schedules: decay interval: same start (green), same end (red), partition (yellow), overlapped (blue); decay power law: homogeneous (continuous), progressive (dotted).
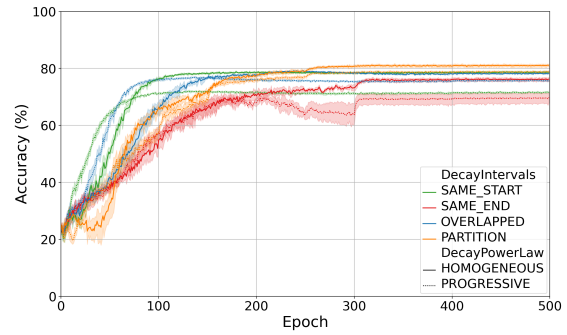
(a)
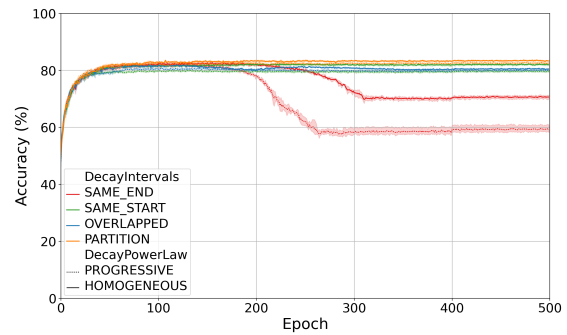
(b)

Figure 3. Performance of ANA using dynamic noise schedules under the uniform noise type and different forward computation strategies: random 3a, mode 3b. Each plot reports multiple schedules: decay interval: same start (green), same end (red), partition (yellow), overlapped (blue); decay power law: homogeneous (continuous), progressive (dotted).

ers. In particular, we observed a remarkable correspondence between the predictions of Theorem 2 and the empirical performance of different dynamic noise schedules. Indeed, delaying the annealing of the regularising distributions in the earlier layers with respect to the later layers results in huge accuracy degradations. Note that when the noise distributions are annealed to Dirac's deltas, local gradients evaluate to zero; therefore, gradient computation can be stopped whenever it reaches a quantiser with annealed noise since its output will not contribute to any update. This observation implies that gradient propagation towards upstream nodes can be interrupted early during training when using the effective partition decay intervals strategy. Consequently, ANA can be used to reduce the computational cost of the backward pass in QNN training, with possible benefits for on-chip training.

References

[1] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. Van Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*. Neural Information Processing Systems (NIPS), 2017. 2

[2] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation, 2013. 1

[3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, V. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. M. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*. Neural Information Processing Systems, 2020. 1

[4] J. Choi, S. Venkataramani, V. Srinivasan, K. Gopalakrishnan, Z. Wang, and P. Chuang. Accurate and efficient 2-bit quantized neural networks. In *Proceedings of the 2nd Conference on Systems and Machine Learning (SysML 2019)*. Conference on Systems and Machine Learning, 2019. 1, 5

[5] L. Deng, P. Jiao, J. Pei, Z. Wu, and G. Li. GXNOR-Net: training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework. *Neural Networks*, 100:49–58, 2018. 2, 3

[6] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha. Learned step size quantization. In *Proceedings of the 2020 International Conference on Learning Representations (ICLR 2020)*. International Conference on Learning Representations (ICLR), 2020. 1, 5

[7] S. Han, H. Mao, and W. J. Dally. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding. In *Proceedings of the 2016 International Conference on Learning Representations (ICLR 2016)*. International Conference on Learning Representations (ICLR), 2016. 1

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2016)*. IEEE, 2016. 1

[9] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016)*. Neural Information Processing Systems (NIPS), 2016. 1, 4

[10] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Quantized neural networks: training neural networks with low precision weights and activations. *Journal of Machine Learning Research*, 18:1–30, 2018. 1

[11] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2018)*. IEEE, 2018. 1

[12] S. R. Jain, A. Gural, M. Wu, and C. H. Dick. Trained quantization thresholds for accurate and efficient fixed-point inference of deep neural networks. In *Proceedings of the 3rd Conference on Machine Learning and Systems (MLSys 2020)*. Conference on Machine Learning and Systems, 2020. 1

[13] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, pages 455–500, 2009. 1

[14] A. Krizhevsky, V. Nair, and G. E. Hinton. https://www.cs.toronto.edu/~kriz/cifar.html, 2014. 5

[15] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein. Training quantized nets: a deeper understanding. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*. Neural Information Processing Systems (NIPS), 2017. 2

[16] X. Lin, C. Zhao, and W. Pan. Towards accurate binary convolutional neural networks. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017)*. Neural Information Processing Systems (NIPS), 2017. 1

[17] Z. Liu, B. Wu, W. Luo, X. Yang, W. Liu, and K.-T. Cheng. Bi-real net: enhancing the performance of 1-bit CNNs with improved representational capability and advanced training algorithm. In *2018 European Conference on Computer Vision (ECCV)*. Springer, 2018. 1, 2, 3

[18] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*. MLResearchPress, 2010. 1

[19] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*

2011. Neural Information Processing Systems (NIPS), 2011. 14

[20] W. Severa, C. M. Vineyard, R. Dellana, S. J. Verzi, and J. B. Aimone. Training deep neural networks for binary communication with the Whetstone method. *Nature Machine Intelligence*, 1:86–94, 2019. 2, 3, 4

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*. ICLR, 2015. 5

[22] M. Tan and Q. V. Le. EfficientNet: rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*. MLResearchPress, 2019. 1

[23] M. Tan, R. Pang, and Q. V. Le. EfficientDet: scalable and efficient object detection. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2020)*. IEEE, 2020. 1

[24] M. van Baalen, C. Louizos, M. Nagel, R. Ali Amjad, Y. Wang, T. Blankevoort, and M. Welling. Bayesian bits: unifying quantization and pruning. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*. Neural Information Processing Systems (NIPS), 2020. 1

[25] P. Warden. Speech Commands: a dataset for limited-vocabulary speech recognition, 2018. 14

[26] J. Wu, L. Deng, G. Li, J. Zhu, and L. Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:1–12, 2018. 2, 3

[27] P. Yin, J. Lyu, S. Zhang, S. Osher, Y. Qi, and J. Xin. Understanding straight-through estimator in training activation quantized neural nets. In *Proceedings of the 2019 International Conference on Learning Representations (ICLR 2019)*. International Conference on Learning Representations (ICLR), 2019. 2, 4

[28] X. Zhang, J. Zou, K. He, and J. Sun. Accelerating very deep convolutional networks for classification and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016. 1

[29] Y. Zhang, N. Suda, L. Lai, and V. Chandra. Hello Edge: keyword spotting on microcontrollers, 2017. 14

[30] B. Zhuang, C. Shen, M. Tan, L. Liu, and I. Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*. IEEE, 2019. 1