Bayesian Sparsification of Deep $\mathbb{C}$-valued Networks

Ivan Nazarov [1]   **Evgeny Burnaev** [1]

Abstract

With continual miniaturization ever more applications of deep learning can be found in embedded systems, where it is common to encounter data with natural representation in the complex domain. To this end we extend Sparse Variational Dropout to complex-valued neural networks and verify the proposed Bayesian technique by conducting a large numerical study of the performance-compression trade-off of $\mathbb{C}$-valued networks on two tasks: image recognition on MNIST-like and CIFAR10 datasets and music transcription on MusicNet. We replicate the state-of-the-art result by Trabelsi et al. (2018) on MusicNet with a complex-valued network compressed by $50 - 100\times$ at a small performance penalty.

1. Introduction

Deep neural networks are an integral part of machine learning and data science toolset for practical data-driven problem solving. With continual miniaturization ever more applications can be found in embedded systems. Common embedded applications include on-device image recognition and signal processing. Despite recent advances in generalization and optimization theory specific to deep networks, deploying in actual embedded hardware remains a challenge due to storage, real-time throughput, and arithmetic complexity restrictions (Han et al., 2015). Therefore, compression methods for achieving high model sparsity and numerical efficiency without losing much in performance are especially relevant.

Complex-valued nature of the data in acoustic and radio signal processing has been the main driver behind the adoption of $\mathbb{C}$-valued neural networks ($\mathbb{C}$VNN). Hirose (2009) argues that the combined phase-magnitude effect of $\mathbb{C}$-valued transformations removes the excess degrees of freedom,

that cause degenerate transformations in $\mathbb{R}$-valued networks with twice the feature dimensions. Their study demonstrates superiority of $\mathbb{C}$VNN in landmine detection using ground penetrating radar imaging. Other examples, where $\mathbb{C}$-valued networks have outperformed $\mathbb{R}$-valued networks, include magnetic resonance (Hui and Smith, 1995; Wang et al., 2020) and radar imaging (Haensch and Hellwich, 2010; Zhang et al., 2017), music transcription and spectral speech modelling (Trabelsi et al., 2018; Wisdom et al., 2016), and wireless signal classification (Yang et al., 2020). Tarver et al. (2019) have lowered the out-of-band power leakage with a $\mathbb{C}$-valued network for digital signal predistortion. The networks have also been applied to non-$\mathbb{C}$-valued domains, such as image classification (Popa, 2017), sequence modelling (Danihelka et al., 2016), and motion prediction (Wolter and Yao, 2018), and for stabilizing backpropagation in RNN (Wisdom et al., 2016).

Despite promising results for embedded signal processing applications, $\mathbb{C}$-valued networks remain a niche in deep learning, and as such little attention has been paid to compression methods specific to $\mathbb{C}$VNN. Yet there is an abundance of research related to real-valued network compression, and many results can be applied to $\mathbb{C}$VNN. Methods such as knowledge distillation (Hinton et al., 2015), which trains a small network to replicate a large well-trained teacher, low-rank matrix (Denton et al., 2014) and tensor decomposition (Novikov et al., 2015), or magnitude-based parameter pruning (Zhu and Gupta, 2018) can be adapted to $\mathbb{C}$VNN without modifications. Parameter quantization and conversion from floating to fixed point arithmetic (Courbariaux et al., 2015; Uhlich et al., 2020), appear to be readily applicable as well. For example, Wu et al. (2019) adapt k-means quantization to $\mathbb{C} \simeq \mathbb{R}^2$ parameters and successfully compress $\mathbb{C}$VNN with the "prune-quantize-code" procedure of Han et al. (2016).

Other methods cannot be translated to $\mathbb{C}$VNN this straightforwardly. Probabilistic ℓ_0 regularization of Louizos et al. (2018) prune networks using multiplicative $[0, 1]$-valued stochastic masks with distributions having an atom at 0, yet differentiable via the reparameterization trick (Kingma and Welling, 2014). By sharing a single mask value within a group of parameters their approach can be adapted to $\mathbb{C}$ parameters. However, methods such as Hessian-based parameter pruning (LeCun et al., 1990) or Sparse Variational

[1] Centre for Data Intensive Sciecne and Engineering, Skolkovo Insitiute of Science and Technology, Msocow, Russia . Correspondence to: Ivan Nazarov <ivan.nazarov@skolkovotech.ru>.

Dropout (Molchanov et al., 2017) require additional considerations.

Gale et al. (2019) compare magnitude pruning, ℓ_0 regularization and Sparse Variational Dropout (VD) on large-scale models. Their results suggest that VD may achieve good accuracy-sparsity balance and outperform pruning and ℓ_0 in deep architectures, although pruning is preferred for simplicity, stability and speed. They also observe that VD induces non-uniform sparsity throughout the model, which He et al. (2018) have shown to be essential for superior compression.

Sparse Variational Dropout is a Bayesian Variational Inference method with automatic parameter relevance determination effect. In this study we extend *Sparse VD* to $\mathbb{C}$VNN, inspired by the results of Gale et al. (2019), and motivated by seldom application of Bayesian Inference to $\mathbb{C}$-valued networks (Popa, 2017) and apparent scarcity of compression methods specific to them. We assess the performance-compression trade-off of the extension by conducting a large-scale numerical study on image classification on MNIST-like and CIFAR10 datasets and music transcription on MusicNet.

The paper is structured as follows. Sec. 2 reviews Variational Dropout, and sec. 3 provides a brief summary of the inner workings of complex-valued networks. The main contribution of this study is presented in sec. 4, where we provide the details of $\mathbb{C}$-valued variational sparsification methods. In sec. 5 we estimate the compression and performance trade-off on shallow and deep $\mathbb{C}$-valued networks, and discuss the outcomes.

2. Variational Dropout

2.1. Variational Inference

In broad terms Bayesian Inference is a principled framework for reasoning about uncertainty and updating prior beliefs about model's parameters in accordance with evidence or empirical data into a posterior distribution. The posterior is useful for inference regarding unobserved data, predictive statistics, parameter confidence regions, and model's uncertainty.

For an observed dataset $D = (x_i)_{i=1}^N$ and statistical model $p(D \mid \omega) = \prod_i p(x_i \mid \omega)$ with parameters ω the Bayes rule transforms prior hypotheses $\pi(\omega)$ about the unknown distribution of model's parameters into the posterior distribution: $p(\omega \mid D) = \frac{p(D|\omega)\pi(\omega)}{p(D)}$. Save for the relatively simple set-ups, either the posterior distribution itself or the mathematical expectations it is involved in are analytically intractable or impractical to compute numerically. Variational Inference (VI), proposed by Jordan et al. (1999), can be used in such cases to make approximate inference. The approach finds an approximation within some distri-

bution family $q_\theta(\omega)$, which is closest to the true posterior distribution in terms of Kullback-Leibler divergence: $KL(q_\theta(\omega)\|p(\omega \mid D)) = \mathbb{E}_{\omega \sim q_\theta} \log \frac{q_\theta(\omega)}{p(\omega|D)}$. Jordan et al. (1999) show that this problem is equivalent to variational *maximization* of the *Evidence Lower Bound* (ELBO)

$$\mathcal{L}(\theta; \lambda) = -KL(q_\theta\|\pi_\lambda) + \sum_{i=1}^N \mathbb{E}_{\omega \sim q_\theta} \log p(x_i \mid \omega), \quad (1)$$

where the variational parameters θ and λ parameterize the approximation and the prior, respectively. Kullback-Leibler and, by proxy, ELBO are standard objectives in VI, however it is possible to use other objectives, provided the true posterior $p(\omega \mid D)$ is evaluated only through $\log p(D \mid \omega)$ and $\log \pi(\omega)$, (Ranganath et al., 2016).

In subsequent years several improvements to Variational Inference approach were introduced. To make VI able to handle large-scale datasets Hoffman et al. (2013) proposed *Stochastic Variational Inference*, which uses stochastic gradient optimization of (1) based on noisy unbiased gradient estimates of ELBO computed on random mini-batches from the dataset. Titsias and Lázaro-Gredilla (2014) translated the dependence on location–scale parameters of q_θ to the function inside its expectation and proposed *Doubly Stochastic Variational Inference*. DSVI constructs an unbiased finite-sample estimator of the gradient of (1) by both subsampling the dataset and sampling from q_θ, without forfeiting convergence of SVI.

Independently, Kingma and Welling (2014) proposed *Stochastic Gradient Variational Bayes*, which is an alternative efficient doubly stochastic estimator applicable to models with parameters ω that are continuous random variables amenable to the *reparameterization trick*, i.e. $\omega \sim q_\theta(\omega)$ is equivalent in distribution to $\omega = g_\theta(\varepsilon)$ for some nonparametric random variable $\varepsilon \sim p(\varepsilon)$ and $g(\varepsilon; \theta)$ differentiable with respect to θ. The estimator of (1) with L reparameterized draws per element in the mini-batch of size M is given by

$$\widetilde{\mathcal{L}}(\theta; \lambda) = -KL(q_\theta\|\pi_\lambda) + \frac{N}{ML} \sum_{k,l} \log p(x_{i_k} \mid g(\varepsilon_{lk}; \theta)),$$

$$(2)$$

where $(x_{i_k})_{k=1}^M$ is a random subsample from D and $(\varepsilon_{lk})_{l=1}^L$ are $k = 1..M$ independent iid samples from p_ε. Figurnov et al. (2018) extended the scope of the reparameterization gradients to include continuous distributions such as Gamma and von Mises. To handle the case of non-reparameterizable ω in doubly stochastic VI, e.g. discrete random parameters, Titsias and Lázaro-Gredilla (2015) proposed *local expectation gradients*, which is a version of REINFORCE gradient estimator (Williams, 1992) with variance reduced by careful use of dependence structures in the model.

In this study we use the SGVB estimator (2) with $L = 1$

and the *local reparameterization trick* proposed by Kingma et al. (2015). They argued that this gradient estimator can be made more statistically and computationally efficient, if the structure of the model permits translating global stochasticity of ω down to local intermediate states of computation. The class of models that allow this include non-recurrent computational graphs, exemplified by neural networks with parameters $\omega \sim q_\theta$. In their case, (2) would require that the *entire* set of network's parameters ω be independently drawn for each element in the mini-batch. Since that the parameters in a network naturally split into subsets with non-overlapping layer-wise effects, it is standard to assume that the approximation $q_\theta(\omega)$ is factorized over layers. Furthermore, if $W \in \mathbb{R}^{n \times m}$ in a linear layer $y = b + W^\top x$ with $q_\theta(W) = \prod_{ij} \mathcal{N}(w_{ij} | \mu_{ij}, \sigma_{ij}^2)$, then by virtue of y being a linear transformation of W, we get

$$q(y) = \prod_i \mathcal{N}\Big(y_i \ b_i + \sum_j \mu_{ij} x_j, \sum_j \sigma_{ij}^2 x_j^2\Big). \quad (3)$$

This yields outputs equivalent in distribution to sampling W for each element in the mini-batch, which produces the SGVB estimator with smaller variance, as demonstrated by Kingma et al. (2015).

2.2. Dropout

Variational Inference can be used as model regularization and sparsification method for certain posterior approximation q_θ and prior π.

Dropout, proposed by Hinton et al. (2012), prevents overfitting by injecting multiplicative binary noise into layer's weights, which breaks up co-adaptations that could occur during training. Wang and Manning (2013) argued that the overall effect of binary Dropout on the intermediate outputs can be approximated by a Gaussian with weight-input dependent mean and variance via the Central Limit Theorem. Srivastava et al. (2014) proposed using independent $\mathcal{N}(1, 1)$ multiplicative noise, arguing that higher entropy of a Gaussian has better regularizing effect. Gal and Ghahramani (2016) showed that Dropout is a Bayesian approximation method with close ties to deep Gaussian Processes that yields inexpensive model uncertainty estimates. In a study concerning multitask learning Cheung et al. (2019) demonstrated the possibility of storing task-specific parameters in non-destructive superposition within a single network. Regarding Dropout their argument implies that if the single task setting is viewed as multitask learning with replicated task, then by sampling uncorrelated binary masks Dropout acts as a superposition method, utilizing the learning capacity of the network better.

Kingma et al. (2015) provided a unifying perspective on Dropout, DropConnect (Wan et al., 2013), and Gaussian Dropout (Wang and Manning, 2013) through the lens of Vari-

ational Inference and propose *Variational Dropout*. They argued that the multiplicative noise introduced by Dropout methods induces a distribution equivalent to a fully factorized variational posterior $q_\theta(\omega) = \prod_j q_\theta(\omega_j)$, where $q_\theta(\omega_j)$ is $\omega_j = \mu_j \eta_j$ with $\eta_j \sim p_\theta(\eta_j)$ iid from some $p_\theta(\eta)$.

Variational Dropout uses *fully factorized Gaussian* approximation $q_\theta(\omega) = \prod_j \mathcal{N}(\omega_j | \mu_j, \alpha_j \mu_j^2)$ and factorized scale invariant log-uniform prior $\pi(\omega)$ with $\pi(\omega_j) \propto |\omega_j|^{-1}$. Molchanov et al. (2017) noticed that α_j reflects the relevance of the parameter ω_j it is associated to by being the ratio of its squared mean to its effective variance. Based on this observation they proposed *Sparse Variational Dropout*, a modification that enables automatic model sparsification by optimizing α_j for each individual parameter. Louizos et al. (2017) extended the idea to structured sparsity by considering hierarchical prior and variational approximation. They grouped the parameters ω_j and coupled them within each one through a shared latent variable, which on the whole enabled pruning entire input features in each layer.

Due to factorization assumption, the term $KL(q_\theta \| \pi)$ in (2) for Sparse VD unravels into $\sum_j K(\frac{\sigma_j^2}{\mu_j^2})$ with

$$K(\alpha) \propto \frac{1}{2} \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,1)} \log \ \frac{1}{\sqrt{\alpha}} + \varepsilon \ ^2. \quad (4)$$

Kingma et al. (2015) approximated $K(\alpha)$ over $\alpha \in (0, 1)$ by a polynomial with a logarithmic term, and later Molchanov et al. (2017) refined the approximation of (4) by weighted sum of a sigmoid and a soft-plus term. In appendix D we verify the derivative of their approximation against a Monte-Carlo estimate for α varying over a fine $\log$-scale grid and the exact expression for gradient of (4).

Kharitonov et al. (2018) addressed theoretical issues with improper prior π in Sparse VD, emphasized by Hron et al. (2018), and proposed *Automatic Relevance Determination* Variational Dropout, by replacing $\pi(\omega_j)$ with a proper Gaussian prior $\pi_\lambda(\omega_j) = \mathcal{N}(\omega_j | 0, \tau_j^{-1})$ with learnable precision $\tau_j > 0$ (Neal, 1996). This recast the VD as the *Empirical Bayes* approach, which performs Bayesian Inference over ω, but uses Maximum Likelihood estimates for the hyperparameters λ, (MacKay, 1994). Maximizing (2) over τ, holding other parameters fixed, yields $\tau_j^* = (\mu_j^2 + \sigma_j^2)^{-1}$, whence

$$K(\alpha) = \frac{1}{2} \log \left(1 + \frac{1}{\alpha}\right). \quad (5)$$

Simultaneously the method Molchanov et al. (2017) proposed to use *additive noise parameterization* in the factorized Gaussian $q_\theta(\omega)$ in conjunction with the local reparameterization trick. They reverted the (μ, α) parameterization in $q_\theta(\omega)$ back to (μ, σ^2), arguing that it reduces the variance of the SGVB (2), by rendering the gradient with respect to μ independent from the local noise, injected by (3). This

modification is important for pruning, since μ of a relevant parameter serves as the estimate of its value.

3. ℂ-valued Networks

ℂ-valued neural networks are networks that rely on the arithmetic in the complex domain. To achieve this implementations of ℂVNN use the geometric representation of a complex number as paired *real* and *imaginary* values, $\mathbb{C} \simeq \mathbb{R}^2$, ensuring that the resulting $\mathbb{R}$-valued computational graph respects ℂ-arithmetic. For example, $f \colon \mathbb{C}^n \to \mathbb{C}^m$ is identified with a real vector-valued function $F \colon \mathbb{R}^{2m} \to \mathbb{R}^{2m}$ defined via $F(u, v) = (\Re f(u + \jmath v), \Im f(u + \jmath v))$, $\Re$ and $\Im$ denoting the real and imaginary parts, respectively. When f is a ℂ-valued linear transformation, the computations are "wired" so that

$$F(u, v) = \begin{pmatrix} Pu - Qv \\ Pv + Qu \end{pmatrix} = \begin{pmatrix} P & -Q \\ Q & P \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}, \quad (6)$$

with $P, Q \colon \mathbb{R}^n \to \mathbb{R}^m$ given by $P = \Re f$ and $Q = \Im f$ restricted to $\mathbb{R}^n$. Non-linearities in ℂVNN can be hyperbolic functions or maps that operate on ℂ numbers in planar form, $z \mapsto \sigma(\Re z) + \jmath\sigma(\Im z)$, or polar form $re^{\jmath\phi} \mapsto \sigma(r, \phi)$.

This $\mathbb{C} \simeq \mathbb{R}^2$ identification allows straightforward retrofitting of ℂVNN into existing $\mathbb{R}$-valued auto-differentiation frameworks for deep learning. This act is backed by Wirtinger (ℂℝ) calculus, which enables generalized treatment of functions of complex argument, by regarding z and its *complex conjugate* $\overline{z}$ as independent variables and defining derivative operators with respect to them through partial derivatives with respect to real and imaginary parts. These definitions simplify manual analysis of ℂ derivatives and satisfy the product and chain rules, respect complex conjugation and linearity for $\mathbb{C} \to \mathbb{C}$ maps, and as such were used to define ℂ version of back-propagation, (Benvenuto and Piazza, 1992; Guberman, 2016). However, since auto-differentiation frameworks can algorithmically handle computational graphs of arbitrary complexity, explicit use of Wirtinger derivatives is not required, especially considering the fact that the direction of the steepest ascent of a $\mathbb{C} \to \mathbb{R}$ function is given by complex conjugate gradient $\nabla_{\overline{z}}$, which coincides with the classical gradient of the same function viewed as $\mathbb{R}^2 \to \mathbb{R}$, (see appendix C).

Development of deep ℂ-valued networks has been active. Haensch and Hellwich (2010) put forward ℂ-valued convolutional networks, Guberman (2016) and Popa (2017) developed modifications of pooling, Arjovsky et al. (2016) and Wisdom et al. (2016) proposed ℂ-valued RNNs with unitary recurrent transition matrices, and Danihelka et al. (2016) developed ℂ-valued holographic representations for LSTMs. More recently Trabelsi et al. (2018) proposed ℂ-valued batch-normalization and weight initialization, Wolter and Yao (2018) investigated different ℂ-valued gating mechanisms for RNNs, and Yang et al. (2020) proposed ℂ-valued self-attention and complex transformer architecture. It merits noting that Gaudet and Maida (2018) generalized ℂVNN further to deep quaternion-valued networks, and Vecchi et al. (2020) studied sparsity inducing regularizers for them.

4. ℂ-Variational Dropout

In this section we develop Sparse Variational Dropout for ℂVNN by using a fully factorized complex Gaussian posterior approximation. We outline the ℂ version of the local reparameterization trick and derive the divergence penalties in (2). The proposed ℂ-valued extension can readily be a part of a hierarchical variational approximation for structured sparsity (Louizos et al., 2017).

4.1. ℂ-Gaussian Distribution

A vector $z \in \mathbb{C}^m$ has complex Gaussian distribution, $q(z) = \mathcal{CN}_m(\mu, \Gamma, C)$ with mean $\mu \in \mathbb{C}^m$, complex covariance and relation matrices Γ and C, respectively, if

$$\begin{pmatrix} \Re z \\ \Im z \end{pmatrix} \sim \mathcal{N}_{2m}\left(\begin{pmatrix} \Re \mu \\ \Im \mu \end{pmatrix}, \tfrac{1}{2} \begin{pmatrix} \Re(\Gamma+C) & \Im(C-\Gamma) \\ \Im(\Gamma+C) & \Re(\Gamma-C) \end{pmatrix} \right), \quad (7)$$

provided Γ is positive definite Hermitian matrix, $C^\top = C$, and $\overline{\Gamma} \succeq \overline{C}\Gamma^{-1}C$. Matrices Γ and C are given by $\mathbb{E}(z - \mu)\overline{(z - \mu)^\top}$ and $\mathbb{E}(z - \mu)(z - \mu)^\top$, respectively, and the random vector z is a *circularly symmetric* ℂ-Gaussian vector if z and $\overline{z}$ are uncorrelated, i.e. $C = 0$. The entropy of z terms of Γ and C is

$$\begin{aligned} \mathbb{H}(q) &= -\mathbb{E}_{z \sim q} \log q(z) && (8) \\ &= \tfrac{1}{2} \log \det(\pi e \Gamma) \det(\pi e(\overline{\Gamma} - \overline{C}\Gamma^{-1}C)) \\ &= \log \det(\pi e \Gamma), \quad \text{for } C = 0. \end{aligned}$$

Parameterization of a univariate ℂ-Gaussian distribution is simpler: $\mathcal{CN}(\mu, \sigma^2, \sigma^2\xi)$ with $\xi \in \mathbb{C}$ such that $|\xi| \leq 1$ and $\sigma^2 \geq 0$. By (8) its entropy is $\log \pi e \sigma^2 \sqrt{1 - |\xi|^2}$.

ℂ-Gaussianity is preserved under linear transformations, i.e. for $A \in \mathbb{C}^{n \times m}$ and $b \in \mathbb{C}^n$

$$b + Az \sim \mathcal{CN}_n\big(b + A\mu, A\Gamma\overline{A^\top}, ACA^\top\big). \quad (9)$$

Therefore, if we have a $\mathbb{C}^{n \times m}$ matrix W with *independent* ℂ-Gaussian entries, i.e.

$$q(W) = \prod_{ij} \mathcal{CN}(\mu_{ij}, \Sigma_{ij}, \Sigma_{ij}\xi_{ij}), \quad (10)$$

with $\mu, \xi \in \mathbb{C}^{n \times m}$, $\Sigma \in [0, +\infty)^{n \times m}$ and $|\xi_{ij}| \leq 1$, then for $x \in \mathbb{C}^m$ and $b \in \mathbb{C}^n$ each component y_i of $y = b + Wx$ is independent univariate ℂ-Gaussian

$$y_i \sim \mathcal{CN}\Big(b_i + \sum_{j=1}^m \mu_{ij}x_j, \sum_{j=1}^m \Sigma_{ij}|x_j|^2, \sum_{j=1}^m \Sigma_{ij}x_j^2\xi_{ij}\Big). \quad (11)$$

This is the $\mathbb{C}$-Gaussian version of the local reparameterization trick (3). It requires three matrix-vector operations: $\mathbb{C}$-valued $b + \mu x$ and $C x^2$, and $\mathbb{R}$-valued $\Sigma |x|^2$, where $C_{ij} = \Sigma_{ij} \xi_{ij}$ and the complex modulus and square are applied elementwise. (4.1) can be applied to any layer, the output of which depends linearly on its parameters, such as convolutional, affine, and bilinear transformations $((x, z) \mapsto b_j + x^\top W^{(j)} z)$. Similar to the $\mathbb{R}$ case, $\mathbb{C}$ convolutions draw independent realizations of W for each spatial patch in the input (Molchanov et al., 2017). This provides faster computations and better statistical efficiency of the SGVB gradient estimator by eliminating correlation from overlapping patches (Kingma et al., 2015) and allowing (11) to efficiently leverage $\mathbb{C}$ convolutions of the relation and variance kernels with elementwise complex squares x^2 and amplitudes $|x|^2$.

For $\mathbb{C}$-Sparse Variational Dropout we propose to use fully factorized $\mathbb{C}$-Gaussian approximation (10) with $\xi_{ij} = 0$ and additive noise parameterization ($\alpha_{ij} = \frac{\Sigma_{ij}}{|\mu_{ij}|^2}$) for weights in dense linear, convolutional and other effectively parameter-affine layers. Point estimates are used for biases.

4.2. The priors

For a fully factorized approximation $q(\omega)$ and factorized prior $\pi(\omega) = \prod_{ij} \pi(\omega_{ij})$, the divergence term (2) is

$$KL(q\|\pi) = -\sum_{ij} \mathbb{H}(q(\omega_{ij})) + \mathbb{E}_{q(\omega_{ij})} \log \pi(\omega_{ij}). \quad (12)$$

We consider two fully factorized priors: an improper prior, resembling VD, and $\mathbb{C}$-Gaussian ARD prior. We omit subscripts ij for brevity in this section.

4.2.1. VD PRIOR

From (8) and $\xi = 0$ the KL-divergence for an improper prior $\pi(\omega) \propto |\omega|^{-\beta}$ with $\beta \geq 1$ is

$$KL(q\|\pi) \propto -\log \sigma^2 + \frac{\beta}{2}\left(\mathbb{E}_{\omega \sim q(\omega)} \log|\omega|^2\right). \quad (13)$$

For $\mu \neq 0$ and $\sigma^2 = \alpha|\mu|^2$ property (9) implies $\mathcal{CN}(\mu, \sigma^2, 0) \sim \mu \cdot \mathcal{CN}(1, \alpha, 0)$, whence the expectation in brackets is given by

$$\log \alpha|\mu|^2 + \mathbb{E}_{\varepsilon \sim \mathcal{CN}(0,1,0)} \log \left|\frac{1}{\sqrt{\alpha}} + \varepsilon\right|^2. \quad (14)$$

If $(z_i)_{i=1}^m \sim \mathcal{CN}(0,1,0)$ iid and $\theta \in \mathbb{C}^m$, then $\sum_i |\theta_i + z_i|^2 \sim \chi_{2m}^2(s^2)$ with $s^2 = \sum_i |\theta_i|^2$, i.e. a non-central χ_{2m}^2 with parameter s^2. Its log-moments for general integer $m \geq 1$ have been derived by Lapidoth and Moser (2003, p. 2466). In particular, for $m = 1$ and $\theta \in \mathbb{C}$ we have

$$\mathbb{E}_{z \sim \mathcal{CN}(0,1,0)} \log|\theta + z|^2 = \log|\theta|^2 - Ei(-|\theta|^2), \quad (15)$$

where $Ei(x) = \int_{-\infty}^x t^{-1} e^t dt$ for $x < 0$ is the Exponential Integral, which satisfies $Ei(x) \leq \log(-x)$, $Ei(x) \approx \log(-x) - \gamma$ as $x \to 0$ (γ is Euler's constant) and $Ei(x) \geq -e^x$ for $x \leq -1$. Although Ei is an intractable integral, requiring numerical approximations to compute, its derivative is exact: $\frac{d}{dx} Ei(x) = \frac{e^x}{x}$ at $x < 0$.

From (14) and (15), the terms of the divergence that depend on the parameters are given by

$$KL(q\|\pi) \propto \frac{\beta-2}{2} \log|\mu|^2 + \log \frac{1}{\alpha} - \frac{\beta}{2} Ei(-\frac{1}{\alpha}). \quad (13')$$

We set $\beta = 2$ to make the divergence term depend only on α and add γ so that the right-hand side is non-negative (Lapidoth and Moser, 2003, eq.(84)). Since $Ei(x)$ has simple analytic derivative and (2) depends additively on (13'), it is possible to back-propagate through the divergence without forward evaluation, which speeds up gradient updates.

4.2.2. ARD PRIOR

We consider the fully factorized circularly symmetric $\mathbb{C}$-Gaussian ARD prior $\pi_\tau(\omega) = \mathcal{CN}(\omega|0, \tau^{-1}, 0)$ with $\tau > 0$. The per element divergence term in (12) is

$$KL(q\|\pi_\tau) = -1 - \log(\tau\sigma^2) + \tau(\sigma^2 + |\mu|^2). \quad (16)$$

In Empirical Bayes the prior adapts to the observed data, i.e. (2) is optimized w.r.t. τ of each weight's prior. The Maximum Likelihood estimator of τ is given by the minimizer (16), i.e. $\tau^* = (\sigma^2 + |\mu|^2)^{-1}$, thereby giving

$$KL(q\|\pi_{\tau^*}) = \log\left(1 + \frac{|\mu|^2}{\sigma^2}\right) = \log\left(1 + \frac{1}{\alpha}\right). \quad (16')$$

Thus in both $\mathbb{R}$ and $\mathbb{C}$ cases ARD produces a tractable analytic expression for the KL-divergence term in (2).

4.2.3. $\mathbb{C}$-VARIATIONAL DROPOUT VIA $\mathbb{R}$-SCALING

We consider the following parameterization of W: $W_{ij} = \mu_{ij} \varepsilon_{ij}$, $\varepsilon_{ij} \in \mathbb{R}$ with $\varepsilon_{ij} \sim \mathcal{N}(1, \alpha_{ij})$, yet $\mu \in \mathbb{C}^{n \times m}$. This case corresponds to inference regarding multiplicative noise ε rather than the parameters themselves. Under this parameterization $q(W_{ij})$ is effectively degenerate univariate $\mathbb{C}$-Gaussian (10) with $\Sigma_{ij} = \alpha_{ij}|\mu_{ij}|^2$ and $\xi_{ij} = e^{j\phi_{ij}}$ with $\phi_{ij} = \arg \mu_{ij}$, thereby making the complex relation parameter in (11) equal $\sum_j \alpha_{ij}(x_{ij}\mu_{ij})^2$, which is non-zero. The KL-divergence term coincides with (4), however the major drawback of this approximation is that the gradient of the loss with respect to μ cannot be disentangled from the local output noise by additive reparameterization.

5. Experiments

To verify the proposed $\mathbb{C}$-valued variational sparsification methods presented above and explore their compression-performance trade-off we carry out a numerical study of $\mathbb{C}$VNN for image classification and music transcription.

Since image data is not naturally ℂ-valued, we preprocess it using the natural inclusion $\mathbb{R} \hookrightarrow \mathbb{C}$ (`raw`, $\Im z = 0$) or applying the two-dimensional Fourier Transform (`fft`), centering the lower frequencies. We do not train an auxiliary network that synthesizes the imaginary component from $\mathbb{R}$ input data (Trabelsi et al., 2018). Following Wolter and Yao (2018) and Trabelsi et al. (2018), the class logit scores are taken as real part of the complex-valued output of a network.

The networks are trained in three successive stages in every experiment: the *"pre-train"* stage for pre-training the network, the *"sparsify"* stage to determine parameter relevance using Variational Dropout, and the *"fine-tune"* to train the pruned network (sec. 5.1). Network's parameters are initialized with values from the previous stage. Networks are trained with ADAM optimizer, with the learning rate reset to 10^{-3} before each stage and global ℓ_2-norm gradient clipping at 0.5.

Each experiment is replicated *five* times to account for random effects from initialization, stochastic gradient optimization, noisy output from intermediate layers, and non-determinism of computations on GPU.

The compression rate is calculated based on the number of floating point values needed to store the network and equals $\frac{n_{\text{par}}}{n_{\text{par}} - n_{\text{zer}}}$, where n_{zer} is the number of explicit zeros at the *"fine-tune"* stage and n_{par} is the total number of values. In a $\mathbb{R}$-valued network each parameter counts as *one* value and as *two* values in a ℂVNN. Each model has a compression limit, determined by biases, shift and scaling in $\mathbb{R}$- and ℂ-valued batch normalization layers.

5.1. Stagewise training

At the *"pre-train"* stage every network is fit "as-is" using deterministic layers and only the likelihood term from (2).

During the *"sparsify"* stage we make every layer stochastic and apply variational sparsification (sec. 4.2.1, 4.2.2, or their $\mathbb{R}$ versions). We inject a coefficient $C \in (0, 1]$ at the KL divergence term in (2):

$$-\frac{C}{N} KL(q_\theta \| \pi_\lambda) + \frac{1}{M} \sum_{k=1}^{M} \log p_\phi(x_{i_k} \mid g(\varepsilon_k; \theta)). \quad (2')$$

In contrast to (Molchanov et al., 2017), who anneal C from zero to one during training, we use constant C and vary it between runs. This allows us to explore the compression-performance profile by balancing model's likelihood and posterior's penalty for diverging form the sparsifying prior in (2'). In particular, higher C implies higher sparsity.

Between *"sparsify"* and *"fine-tune"* stages we compute masks of non-zero weights in each layer based on the relevance scores α (sec. 2.2). Since q_θ factorizes into univariate distributions, a ℂ or $\mathbb{R}$ parameter is considered non-zero

iff $\log \alpha \le \tau$ for $\alpha = \frac{\sigma^2}{|\mu|^2}$. The threshold τ is picked so that the remaining non-zero parameters are within δ relative tolerance of their mode with high probability under the approximate posterior. For a univariate $\mathbb{R}$- or a circularly symmetric ℂ-Gaussian random variable w, $\frac{k|w-\mu|^2}{\alpha|\mu|^2}$ is χ_k^2 distributed with $k = 1$ ($\mathbb{R}$) or 2 (ℂ). For a tolerance $\delta = 50\%$ values $\log \alpha$ below -2.5 yield at least 90% chance of a non-zero $\mathbb{R}$/ℂ parameter. We pick $\tau = -\frac{1}{2}$ to retain parameters sufficiently concentrated around their mode and encourage higher sparsity, at the same time being aware that q_θ is merely an approximation. In comparison, $\tau = 3$ is commonly used as the threshold (Kingma et al., 2015; Molchanov et al., 2017).

At the *"fine-tune"* stage the network reverts back to deterministic architecture and proceeds the same way as the "pre-train" stage, except for training only those parameters, which are specified by sparsity masks.

5.2. MNIST-like datasets

We conduct a moderately sized experiment on MNIST-like datasets of 28×28 greyscale images to study the performance-compression trade-off of the proposed ℂ-valued Sparse Variational Dropout: MNIST (Lecun et al., 1998), KMNIST (Clanuwat et al., 2018), EMNIST (Cohen et al., 2017) and Fashion-MNIST (Xiao et al., 2017). We deliberately use a *fixed* random subset of *ten thousand* images from the train split of each dataset to fit the networks and measure the performance with classification *accuracy* score on the usual test split.

We consider two simple architectures in this experiment, which have been chosen for the purpose of illustrating the compression and understanding the effects of experiment parameters. *TwoLayerDenseModel* is a wide dense ReLU network $784 \rightarrow 4096 \rightarrow n_{\text{out}}$, and *SimpleConvModel* is a ReLU net with two $2d$ $k_5 s_1$ convolutions with filters $20 \rightarrow 50$, two $k_2 s_2$ average pooling steps, and a classifier head $800 \rightarrow 500 \rightarrow n_{\text{out}}$. For each dataset we experiment with all combinations of model kinds ($\mathbb{R}$ or ℂ) and sparsification methods (VD or ARD). To take into account potential differences in the capacity of ℂVNN we consider halving or doubling the number of features in the intermediate layers (Mönning and Manandhar, 2018). Halved ℂVNN are tagged $\frac{1}{2}\mathbb{C}$, and doubled $\mathbb{R}$-valued networks are labelled $2\mathbb{R}$. For `fft` we compare $\{\mathbb{R}, \mathbb{C}, 2\mathbb{R}\}$ and for `raw` – $\{\frac{1}{2}\mathbb{C}, \mathbb{R}, \mathbb{C}\}$.

Stages (sec. 5.1) last for 40, 75 and 40 epochs, respectively, in each experiment. The sparsification threshold τ is fixed at $-\frac{1}{2}$, the training batch size is set to 128 and the base learning rate 10^{-3} is reduced after the 10-th epoch to 10^{-4} at every stage. We vary $C \in \{\frac{3}{2} 2^{-\frac{k}{2}} : k = 2, \cdots, 38\}$ in (2') and repeat each experiment 5 times to get a sample of compression-accuracy pairs.
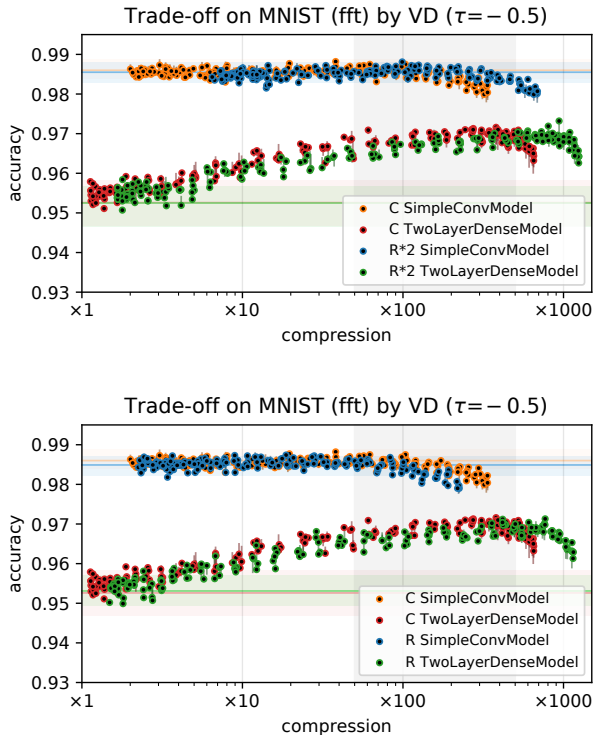
Figure 1. The compression-accuracy curve (VD, `fft`, MNIST): $2\mathbb{R}/\mathbb{C}$ (*top*) and $\mathbb{R}/\mathbb{C}$ (*bottom*).
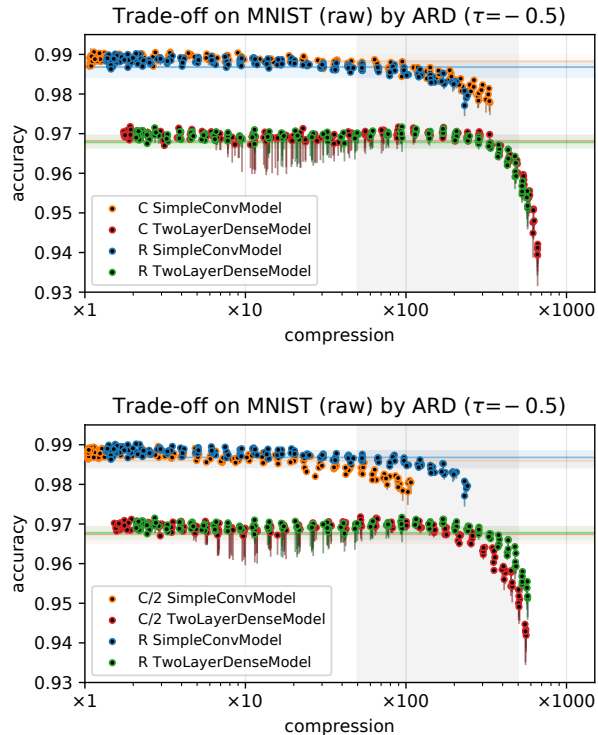

Figure 2. The compression-accuracy curve (ARD, `raw`, MNIST): $\mathbb{R}/\mathbb{C}$ (*top*) and $\mathbb{R}/\frac{1}{2}\mathbb{C}$ (*bottom*).

Figures 1 and 2 depict the resulting compression-accuracy trade-off on MNIST for the models described above. Each point represents the trade-off of the compressed network after fine-tuning, while its tail illustrates the impact of this stage on the performance. Transparent horizontal bands on each plot represent min-max performance spread of the pre-trained uncompressed network on the test split. Results for other MNIST-like dataset are presented appendix A.

The overarching conclusion from the conducted experiments is that both ℂ-ARD and ℂ-VD methods compress similarly to each other, but for the same value of C in (2') ARD yields marginally lower compression and slightly higher performance post fine-tuning. For each fixed C the compression rates after "sparsify" stage are roughly identical. At the same time, "fine-tune" stage almost always improves performance in high compression regime ($\times 50+$ high C in (2')), likely due to regularization from high sparsity. Fourier features catch up to the raw data in terms of performance at high compression rates $\times 100+$ only for the *TwoLayerDenseModel*. Comparison of $\mathbb{R}$ and $\mathbb{C}$ networks with matching architecture, i.e. same effective layer widths in $2\mathbb{R}$ vs. $\mathbb{C}$ and $\mathbb{R}$ vs. $\frac{1}{2}\mathbb{C}$, shows that doubled $\mathbb{R}$ networks perform and compress better than $\mathbb{C}$, due to higher intrinsic redundancy unchecked by $\mathbb{C}$-arithmetic constraint.

5.3. CIFAR10

Having verified the ℂ variational sparsification method on MNIST-like datasets and simple models, we turn to the CIFAR10 dataset comprising 32×32 colour images of 10 classes (Krizhevsky, 2009) and focus on the VGG16 network (Simonyan and Zisserman, 2015). We train the VGG16 network and its ℂ variant, in which we have replaced $\mathbb{R}$-valued layers with their $\mathbb{C}$-valued counterparts. We do not halve or double the features in any network, since the goal of this experiment is to assess the trade-off for a deep convolutional network. Unlike experiment in sec. 5.2, we consider the raw features only, use full training split, measure accuracy on the usual test split, and allocate 20, 40, and 20 epochs to each stage. During training every mini-batch of 128 samples is augmented by random horizontal flipping and random cropping, which is done by zero-padding the image with four pixels and extracting a 32×32 patch from the 40×40 intermediate image.

The compression-accuracy curve in figure 3, constructed for $C = \frac{3}{2} 2^{-\frac{k}{2}}$ with $k = 7, \cdots, 15$, shows that it is possible to confidently achieve around $\times 100$ compression of a deep ℂVNN without losing accuracy, provided the network is fine-tuned after undergoing Variational sparsification. Re-
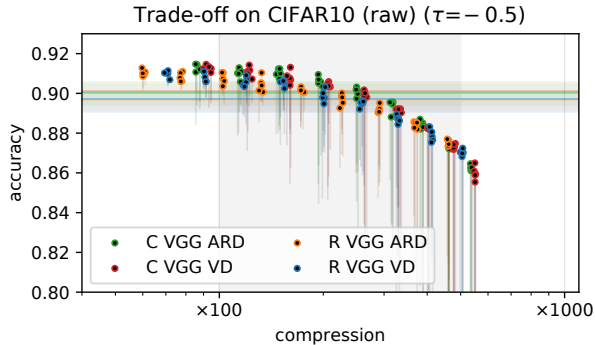
Figure 3. The compression-accuracy profile for the $\mathbb{R}$ and $\mathbb{C}$ VGG16.

Figure 4. Performance-compression curve for **VD**, **ARD**, and the k_3 version compressed with VD.

garding methods themselves, $\mathbb{C}$-VD and $\mathbb{C}$-ARD follow the same declining compression-accuracy pattern, but for the same setting C the latter provides slightly less compression with marginally better accuracy.

5.4. MusicNet

MusicNet is a corpus of 330 annotated classical music recordings used for learning feature representations for music transcription tasks (Thickstun et al., 2017). Trabelsi et al. (2018) have proposed a $1d$ VGG-like $\mathbb{C}$-valued network that surpassed a similar $\mathbb{R}$-valued network and achieved 72.9% pooled *Average Precision* on this dataset. Recently Yang et al. (2020) have reported 74.2% AP with a $\mathbb{C}$-valued transformer, Thickstun et al. (2018) have achieved 77.3% with a four-layer $\mathbb{R}$-valued network on $\log$-spaced spectrogram, and Draguns et al. (2020) report 78.0% with a residual-shuffle-exchange network.

In this experiment we seek to compress of the $\mathbb{C}$VNN proposed by Trabelsi et al. (2018). The dataset is split into the same validation and test samples and handled identically to their study. The input features are $\mathbb{C}$-valued Fourier transforms of 4096-sample windows from each waveform, and the label vectors are taken from annotations at the middle of the window. Each epoch lasts for 1000 random mini-batches of the musical pieces. However, we deviate from the set-up used by Trabelsi et al. (2018) by clipping ℓ_2 norm of the gradients to 0.05 and shifting the low frequencies of the input to the centre to maintain spatial locality for convolutions.

Experiments with the uncompressed model aimed at replicating the original result have shown that early stopping almost always terminates within the first $10 - 20$ epochs of the 200 epochs used in their study, due to the validation performance peaking at $10 - 15$ epochs and steadily declining afterwards. Thus we opt to use shorter stages: 12, 32 and 50 epochs (sec. 5.1), with early stopping activated only during
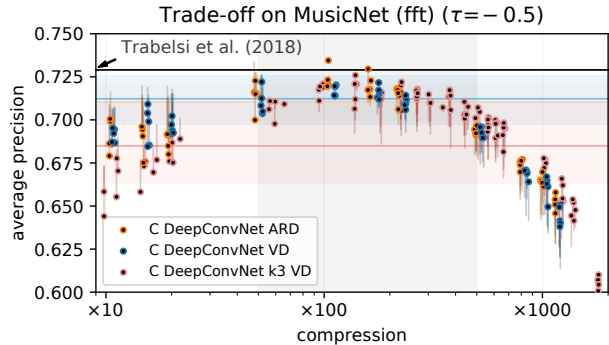
the "fine-tune" stage. To keep the learning rate schedule consistent, we scale the learning rate of 10^{-3} after 5, 10, and 20-th epoch by $\frac{1}{10}$, $\frac{1}{20}$ and $\frac{1}{100}$, respectively.

We explore the $\mathbb{C}$-VD and $\mathbb{C}$-ARD methods by varying C over the grid $\{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\} \cdot 10^{-k}$ with $k = 1, 2, 3$, while keeping τ at $-\frac{1}{2}$. The performance is measured after "pre-train" stage, just before and upon termination of fine-tuning. Additionally, we test the model of Trabelsi et al. (2018), in which we purposefully halve the receptive field of the first convolution from 6 to 3 (denoted by suffix k_3). The motivation is to test if the handicap introduced by the forced compression of the most upstream layer can be alleviated by non-uniform compression, induced by Variational Dropout. We test only $\mathbb{C}$-VD in this sub-experiment, since prior results have not demonstrated significant superiority of one method over another.

The performance-compression frontier in figure 4 shows that VD and ARD deliver similar compression rates, but ARD slightly outperforms in terms of the average precision at the cost of marginally lower compression. At $\times 100$ compression level the k_3 model outperforms its uncompressed baseline, but yields lower AP score than the full model. In conjunction with post-pruning fine-tuning, both $\mathbb{C}$-valued variational sparsification methods achieve average precision level comparable to the result of Trabelsi et al. (2018) with a network having 50-200 times less parameters.

We take the full models compressed with $C \in \{\frac{1}{20}, \frac{1}{200}\}$ and re-run only the fine-tuning stage for various pruning thresholds $\tau \in \{\frac{k}{2} : k = -8 \cdots + 8\}$. The performance-compression curves depicted in figure 5 are parameterized by decreasing τ from left to right, since models are not re-compressed which makes τ monotonically affect the compression rate. From (2') and the relative positions of the curves it can be concluded that C has a much more substantial impact on the compression profile of each method, than
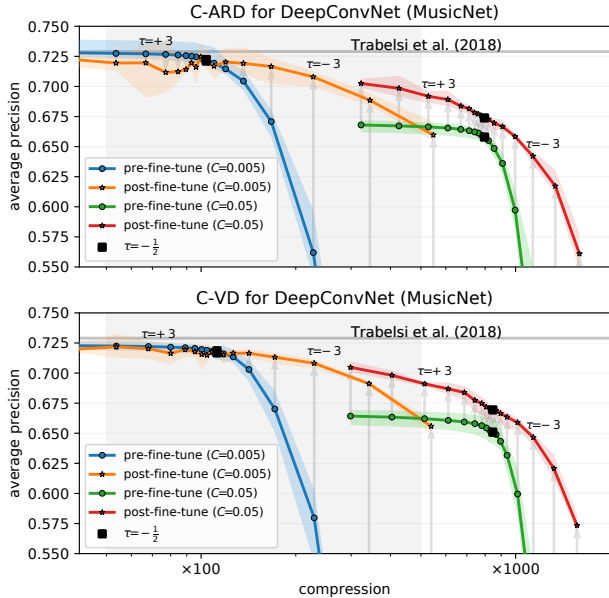
Figure 5. The effect of fine-tuning on performance-compression curves for $C \in \{\frac{1}{20}, \frac{1}{200}\}$ in (2').

Figure 6. Early stopping epoch at fine-tuning stage.

the choice of the pruning threshold.

We provide the following interpretation of the apparent contrast in performance impact borne by fine-tuning between less than $\times 50$ and higher than $\times 100$ compression regimes in figure 4, also observed in sec. 5.2. The value of C in (2') is a good proxy for the ranking of the final compression rate since it directly affects the feedback from sparsifying prior. So, during the 50 epoch allotted for "sparsify" stage, low C prevents the sparsity inducing prior from pulling the posterior sufficiently away from the likelihood-maximizing parameters inherited from the "pre-train" stage. It is reasonable, therefore, to expect that for undercompressed models the fine-tuning stage acts essentially as a continuation of pre-training. And, since we have observed that longer training invariably deteriorates the validation performance, the "fine-tune" stage should lead to overfitting for small C. Figure 6 shows that the models, which have been sparsified with C less than $\frac{1}{400}$, have less than $50\times$ compression and need considerably less training epochs before early stopping terminates the process.

6. Conclusion

In this study we have presented ℂ-valued variational sparsification methods to the ever growing set of tools for learning deep ℂ-valued neural networks. To validate these methods we have carried out a large numerical study of ℂVNN with simple architectures to assess the feasible performance-

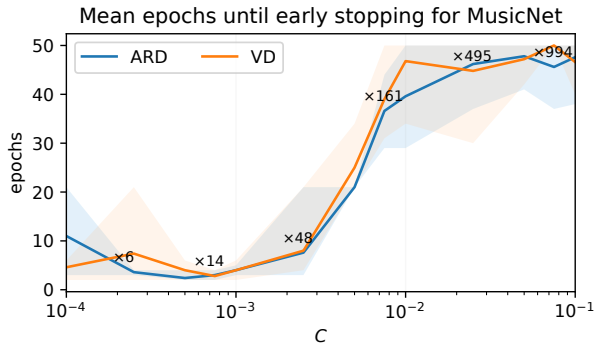compression trade-off, and studied compression of two deep convolutional ℂVNN. At the cost of marginally lower performance, we have achieved $\times 50$-$\times 100$ compression of the deep ℂVNN of Trabelsi et al. (2018) on the MusicNet.

Experimental results show that ℂ-VD (sec. 4.2.1) and ℂ-ARD (sec. 4.2.2) exhibit trade-off profiles matching their ℝ-valued counterparts. This makes us confident that the overall conclusion of Gale et al. (2019) is applicable to ℂVNN and the proposed ℂ-valued variational sparsification methods. Furthermore our findings indicate that between each other under similar circumstances the methods yield comparable compression and performance results, which echoes earlier results by Kharitonov et al. (2018).

This study has direct implications for embedded deep learning applications both in terms of lower storage requirements and higher throughput stemming from fewer floating point multiplications due to sparsity, despite somewhat higher arithmetic complexity of ℂ-valued networks.

Software and Data

The source code for a package based on PyTorch (Paszke et al., 2019), which implements ℂ-valued Sparse Variational Dropout and ARD layers and provides other basic layers for ℂVNN is available at https://github.com/ivannz/cplxmodule. The source code for the experiments and the figures in this study is available at https://github.com/ivannz/complex_paper/tree/v2020.6.

Acknowledgements

References

M. Arjovsky, A. Shah, and Y. Bengio. Unitary Evolution Recurrent Neural Networks. In *International Conference on Machine Learning*, pages 1120–1128, June 2016.

N. Benvenuto and F. Piazza. On the complex backpropagation algorithm. *IEEE Transactions on Signal Processing*, 40(4):967–969, Apr. 1992. ISSN 1053-587X, 1941-0476. doi: 10.1109/78.127967.

B. Cheung, A. Terekhov, Y. Chen, P. Agrawal, and B. Olshausen. Superposition of many models into one. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 10868–10877. Curran Associates, Inc., 2019.

T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. Deep Learning for Classical Japanese Literature. *arXiv:1812.01718 [cs, stat]*, Dec. 2018. doi: 10.20676/00000341. arXiv: 1812.01718.

G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. EMNIST: Extending MNIST to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2921–2926, May 2017. doi: 10.1109/IJCNN.2017.7966217. ISSN: 2161-4407.

M. Courbariaux, Y. Bengio, and J.-P. David. Training deep neural networks with low precision multiplications. *arXiv:1412.7024 [cs]*, Sept. 2015. arXiv: 1412.7024.

I. Danihelka, G. Wayne, B. Uria, N. Kalchbrenner, and A. Graves. Associative Long Short-Term Memory. In *International Conference on Machine Learning*, pages 1986–1994, June 2016. ISSN: 1938-7228 Section: Machine Learning.

E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1269–1277. Curran Associates, Inc., 2014.

A. Draguns, E. Ozoliņš, A. Šostaks, M. Apinis, and K. Freivalds. Residual Shuffle-Exchange Networks for Fast Processing of Long Sequences. *arXiv:2004.04662 [cs, eess]*, Apr. 2020. arXiv: 2004.04662 version: 1.

M. Figurnov, S. Mohamed, and A. Mnih. Implicit Reparameterization Gradients. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 441–452. Curran Associates, Inc., 2018.

Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning*, pages 1050–1059, June 2016. ISSN: 1938-7228 Section: Machine Learning.

T. Gale, E. Elsen, and S. Hooker. The State of Sparsity in Deep Neural Networks. *arXiv:1902.09574 [cs, stat]*, Feb. 2019. arXiv: 1902.09574.

C. J. Gaudet and A. S. Maida. Deep Quaternion Networks. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2018. doi: 10.1109/IJCNN.2018.8489651. ISSN: 2161-4407.

N. Guberman. On Complex Valued Convolutional Neural Networks. *arXiv:1602.09046 [cs]*, Feb. 2016. arXiv: 1602.09046.

R. Haensch and O. Hellwich. Complex-Valued Convolutional Neural Networks for Object Detection in PolSAR data. In *8th European Conference on Synthetic Aperture Radar*, pages 1–4, June 2010.

S. Han, J. Pool, J. Tran, and W. Dally. Learning both Weights and Connections for Efficient Neural Network. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1135–1143. Curran Associates, Inc., 2015.

S. Han, H. Mao, and W. J. Dally. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In Y. Bengio and Y. LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, Lecture Notes in Computer Science, pages 815–832, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01234-2. doi: 10.1007/978-3-030-01234-2_48.

G. Hinton, O. Vinyals, and J. Dean. Distilling the Knowledge in a Neural Network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv:1207.0580 [cs]*, July 2012. arXiv: 1207.0580.

A. Hirose. Complex-valued neural networks: The merits and their origins. In *2009 International Joint Conference on Neural Networks*, pages 1237–1244, June 2009. doi: 10.1109/IJCNN.2009.5178754. ISSN: 2161-4407.

M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013. ISSN 1533-7928.

J. Hron, A. Matthews, and Z. Ghahramani. Variational Bayesian dropout: pitfalls and fixes. In *International Conference on Machine Learning*, pages 2019–2028, July 2018. ISSN: 1938-7228 Section: Machine Learning.

Y. Hui and M. Smith. MRI reconstruction from truncated data using a complex domain backpropagation neural network. In *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing. Proceedings*, pages 513–516, May 1995. doi: 10.1109/PACRIM.1995. 519582.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37(2):183–233, Nov. 1999. ISSN 1573-0565. doi: 10.1023/A:1007665907178.

V. Kharitonov, D. Molchanov, and D. Vetrov. Variational Dropout via Empirical Bayes. *arXiv:1811.00596 [cs, stat]*, Nov. 2018. arXiv: 1811.00596.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

D. P. Kingma, T. Salimans, and M. Welling. Variational Dropout and the Local Reparameterization Trick. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2575–2583. Curran Associates, Inc., 2015.

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

A. Lapidoth and S. M. Moser. Capacity bounds via duality with applications to multiple-antenna systems on flat-fading channels. *IEEE Transactions on Information Theory*, 49(10):2426–2467, Oct. 2003. doi: 10.1109/TIT.2003.817449.

Y. LeCun, J. S. Denker, and S. A. Solla. Optimal Brain Damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 598–605. Morgan-Kaufmann, 1990.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov. 1998. ISSN 1558-2256. doi: 10.1109/5.726791. Conference Name: Proceedings of the IEEE.

C. Louizos, K. Ullrich, and M. Welling. Bayesian Compression for Deep Learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3288–3298. Curran Associates, Inc., 2017.

C. Louizos, M. Welling, and D. P. Kingma. Learning Sparse Neural Networks through L_0 Regularization. Feb. 2018.

D. J. C. MacKay. Bayesian Non-linear Modelling for the Prediction Competition. In *In ASHRAE Transactions, V.100, Pt.2*, pages 1053–1062. ASHRAE, 1994.

D. Molchanov, A. Ashukha, and D. Vetrov. Variational Dropout Sparsifies Deep Neural Networks. In *International Conference on Machine Learning*, pages 2498–2507, July 2017. ISSN: 1938-7228 Section: Machine Learning.

N. Mönning and S. Manandhar. Evaluation of Complex-Valued Neural Networks on Real-Valued Classification Tasks. *arXiv:1811.12351 [cs, stat]*, Nov. 2018. arXiv: 1811.12351.

R. M. Neal. *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer New York, New York, NY, 1996. ISBN 978-0-387-94724-2 978-1-4612-0745-0. doi: 10.1007/978-1-4612-0745-0.

A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. Tensorizing Neural Networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 442–450. Curran Associates, Inc., 2015.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. pages 8026–8037, 2019.

K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Technical University of Denmark, Nov. 2012.

C.-A. Popa. Complex-valued convolutional neural networks for real-valued image classification. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 816–822, May 2017. doi: 10.1109/IJCNN.2017.7965936. ISSN: 2161-4407.

R. Ranganath, D. Tran, J. Altosaar, and D. Blei. Operator Variational Inference. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 496–504. Curran Associates, Inc., 2016.

K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. ISSN 1533-7928.

C. Tarver, A. Balatsoukas-Stimming, and J. R. Cavallaro. Design and Implementation of a Neural Network Based Predistorter for Enhanced Mobile Broadband. In *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*, pages 296–301, Oct. 2019. doi: 10.1109/SiPS47522.2019.9020606. ISSN: 2374-7390.

J. Thickstun, Z. Harchaoui, and S. M. Kakade. Learning Features of Music From Scratch. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

J. Thickstun, Z. Harchaoui, D. P. Foster, and S. M. Kakade. Invariances and Data Augmentation for Supervised Music Transcription. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2241–2245, Apr. 2018. doi: 10.1109/ICASSP.2018.8461686. ISSN: 2379-190X.

M. Titsias and M. Lázaro-Gredilla. Doubly Stochastic Variational Bayes for non-Conjugate Inference. In *International Conference on Machine Learning*, pages 1971–1979, Jan. 2014. ISSN: 1938-7228 Section: Machine Learning.

M. Titsias and M. Lázaro-Gredilla. Local Expectation Gradients for Black Box Variational Inference. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2638–2646. Curran Associates, Inc., 2015.

C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal. Deep Complex Networks. In *International Conference on Learning Representations*, 2018. arXiv: 1705.09792.

S. Uhlich, L. Mauch, F. Cardinaux, K. Yoshiyama, J. A. Garcia, S. Tiedemann, T. Kemp, and A. Nakamura. Mixed Precision DNNs: All you need is a good parametrization. In *International Conference on Learning Representations*, 2020.

R. Vecchi, S. Scardapane, D. Comminiello, and A. Uncini. Compressing deep quaternion neural networks with targeted regularization. *arXiv:1907.11546 [cs, stat]*, Jan. 2020. arXiv: 1907.11546.

L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of Neural Networks using DropConnect. In *International Conference on Machine Learning*, pages 1058–1066, Feb. 2013.

S. Wang and C. Manning. Fast dropout training. In *International Conference on Machine Learning*, pages 118–126, Feb. 2013. ISSN: 1938-7228 Section: Machine Learning.

S. Wang, H. Cheng, L. Ying, T. Xiao, Z. Ke, H. Zheng, and D. Liang. DeepcomplexMRI: Exploiting deep residual network for fast parallel MR imaging with complex convolution. *Magnetic Resonance Imaging*, 68:136–147, May 2020. ISSN 0730-725X. doi: 10.1016/j.mri.2020.02.002.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992. ISSN 1573-0565. doi: 10.1007/BF00992696.

S. Wisdom, T. Powers, J. Hershey, J. Le Roux, and L. Atlas. Full-Capacity Unitary Recurrent Neural Networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4880–4888. Curran Associates, Inc., 2016.

M. Wolter and A. Yao. Complex Gated Recurrent Neural Networks. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 10557–10567, USA, 2018. Curran Associates Inc. event-place: Montréal, Canada.

J. Wu, H. Ren, Y. Kong, C. Yang, L. Senhadji, and H. Shu. Compressing complex convolutional neural network based on an improved deep compression algorithm. *arXiv:1903.02358 [cs]*, Mar. 2019. arXiv: 1903.02358.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747 [cs, stat]*, Sept. 2017. arXiv: 1708.07747.

M. Yang, M. Q. Ma, D. Li, Y.-H. H. Tsai, and R. Salakhutdinov. Complex Transformer: A Framework for Modeling Complex-Valued Sequence. In *ICASSP 2020 - 2020 IEEE*

International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 4232–4236, May 2020. doi: 10.1109/ICASSP40776.2020.9054008. ISSN: 2379-190X.

Z. Zhang, H. Wang, F. Xu, and Y.-Q. Jin. Complex-Valued Convolutional Neural Network and Its Application in Polarimetric SAR Image Classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55 (12):7177–7188, Dec. 2017. ISSN 1558-0644. doi: 10.1109/TGRS.2017.2743222. Conference Name: IEEE Transactions on Geoscience and Remote Sensing.

M. Zhu and S. Gupta. To Prune, or Not to Prune: Exploring the Efficacy of Pruning for Model Compression. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.