Steinmetz Neural Networks for Complex-Valued Data

Shyam Venkatasubramanian
Duke University

Ali Pezeshki
Colorado State University

Vahid Tarokh
Duke University

Abstract

We introduce a new approach to processing complex-valued data using DNNs consisting of parallel real-valued subnetworks with coupled outputs. Our proposed class of architectures, referred to as *Steinmetz Neural Networks*, incorporates multi-view learning to construct more interpretable representations in the latent space. Moreover, we present the *Analytic Neural Network*, which incorporates a consistency penalty that encourages analytic signal representations in the latent space of the Steinmetz neural network. This penalty enforces a deterministic and orthogonal relationship between the real and imaginary components. Using an information-theoretic construction, we demonstrate that the generalization gap upper bound posited by the analytic neural network is lower than that of the general class of Steinmetz neural networks. Our numerical experiments depict the improved performance and robustness to additive noise, afforded by our proposed networks on benchmark datasets and synthetic examples.

1 Introduction

In recent years, the advancement of neural networks has spurred a wealth of research into specialized models designed for processing complex-valued data. Complex-valued neural networks (CVNNs) are a pivotal area of focus due to their intrinsic capability to leverage both the magnitude information and the phase information embedded in complex-valued signals, offering a distinct advantage over their real-valued counterparts (RVNNs) [Hirose, 2012, Guberman, 2016, Trabelsi et al., 2018]. This is crucial across a spectrum of applications including telecommunications, medical imaging, and radar and sonar signal processing [Virtue et al., 2017, Gao et al., 2019, Smith, 2023]. However, CVNNs are often encumbered by higher computational costs and more complex training dynamics [Bassey et al., 2021, Lee et al., 2022, Wu et al., 2023]. These difficulties arise from the necessity to manage and optimize parameters in the complex domain, which can lead to instability in gradient descent methods and challenges in network convergence. Additionally, the search for effective and efficient complex-valued activation functions is a challenge [Scardapane et al., 2020, Lee et al., 2022].

While much of the discussion comparing CVNNs and RVNNs has been focused on the theoretical aspects of CVNNs, the development of improved RVNN architectures for the processing of complex-valued data remains an open problem. We propose to address this problem through a feature learning perspective, leveraging multi-view representation fusion [Sun, 2013, Lahat et al., 2015, Zhao et al., 2017]. By considering the independent and joint information of real and imaginary parts in successive processing steps, we aim to better capture the task-relevant information in complex-valued data.

In response to these considerations, this paper introduces the *Steinmetz Neural Network* architecture, a real-valued neural network that incorporates multi-view learning to improve the processing of complex-valued data for predictive tasks. This architecture aims to mitigate the challenges of training CVNNs while forming more interpretable latent space representations, and comprises separate subnetworks that independently filter the irrelevant information present within the real and imaginary components, followed by a joint processing step. We note that the task-relevant interactions between components are not lost during the separate processing step, as they are handled in joint processing.

An advantage afforded by the Steinmetz neural network's initial separate processing approach is that it provides control over the coherent combination of extracted features before joint processing. This choice is critical, as the proper combination of these features can lead to improved generalization. A key innovation from our approach is the derivation of a consistency constraint that encourages the extracted real and imag-

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258. GitHub repository: `https://tinyurl.com/3zvsh3cf`.

inary features to be related through a deterministic function, which lowers the Steinmetz neural network's generalization upper bound. For practical implementation, we choose this function to be the discrete Hilbert transform, since it ensures orthogonality between the extracted features to increase diversity [Cizek, 1970, Chaudhry et al., 2020]. This methodology, referred to as the *Analytic Neural Network*, attempts to leverage these structured representations to achieve improved generalization over the class of Steinmetz networks.

The organization of this paper is as follows. In Section 2, we review complex and analytic signal representations, and survey the related work from CVNN literature. In Section 3, we present the Steinmetz neural network architecture and discuss its theoretical foundations. In Section 4, we summarize the consistency constraint and provide generalization gap bounds for the Steinmetz network. In Section 5, we introduce the analytic neural network and the Hilbert transform consistency penalty. In Section 6, we present empirical results on benchmark datasets for complex-valued multi-class classification and regression, and provide synthetic experiments. In Section 7, we summarize our work.

Our main contributions are summarized as follows:

1. We introduce the *Steinmetz Neural Network*, a real-valued neural network architecture that leverages multi-view learning to construct more interpretable latent space representations.

2. We propose a consistency constraint on the latent space of the Steinmetz neural network to obtain a smaller upper bound on the generalization gap, and derive these generalization bounds.

3. We outline a practical implementation of this consistency constraint through the Hilbert transform, and present the *Analytic Neural Network*, which promotes analytic signal representations within the latent space of the Steinmetz neural network.

2 Preliminaries

To motivate our framework, we begin by formally defining complex and analytic signal representations, and review existing real-valued neural networks and complex-valued neural networks for complex signal processing. Let $\mathcal{U} = \{0, 1, \ldots, N-1\}$ and $\mathcal{V} = \{0, 1, \ldots, M-1\}$, where $N \in \mathbb{N}$ is the signal period and $M \in \mathbb{N}$ denotes the size of the training dataset, s. To characterize the uncertainty of these signal representations, we denote $X \in \mathbb{C}^{dN}$, $X_R, X_I \in \mathbb{R}^{dN}$ as the features, $Y \in \mathbb{C}^{k}$, $Y_R, Y_I \in \mathbb{R}^{k}$ as the labels, and $Z \in \mathbb{C}^{lN}$, $Z_R, Z_I \in \mathbb{R}^{lN}$ as the latent variables with R, I denoting the respective real and imaginary parts, where:

$$X = (X[0], .., X[N-1]),\ X_R = (X_R[0], ..., X_R[N-1]),$$
$$X_I = (X_I[0], .., X_I[N-1]),\ Z = (Z[0], .., Z[N-1]),$$
$$Z_R = (Z_R[0], .., Z_R[N-1]),\ Z_I = (Z_I[0], .., Z_I[N-1]).$$

Correspondingly, we denote the training dataset using $s = \{(x^m, y^m),\ m \in \mathcal{V}\}$. Let P denote the joint probability distribution of (X, Y), where $(X, Y) \sim P$, and suppose $(x^m, y^m) \overset{\text{i.i.d.}}{\sim} P$. The product measure, $P^{\otimes M}$, posits $S = ((X^0, Y^0), (X^1, Y^1), \ldots, (X^{M-1}, Y^{M-1}))$, where $S \sim P^{\otimes M}$, $s \sim P^{\otimes M}$, and $(X^m, Y^m) \overset{d}{=} (X, Y)$.

2.1 Complex Signal Representation

Consider the complex stochastic process $\{X[n], n \in \mathcal{U}\}$, which comprises the individual real stochastic processes $\{X_R[n], n \in \mathcal{U}\}$ and $\{X_I[n], n \in \mathcal{U}\}$, wherein $X[n] = X_R[n] + iX_I[n]$, $\forall n \in \mathcal{U}$, with $X[n] \in \mathbb{C}^d$. The respective realizations of $\{X_R[n], n \in \mathcal{U}\}$ and $\{X_I[n], n \in \mathcal{U}\}$, denoted by $\{x_R[n], n \in \mathcal{U}\}$ and $\{x_I[n], n \in \mathcal{U}\}$, are real signals. These realizations define the complex signal $\{x[n], n \in \mathcal{U}\}$, wherein $x[n] = x_R[n] + ix_I[n]$, $\forall n \in \mathcal{U}$. This approach is rooted in the foundational work on the complex representation of AC signals [Steinmetz, 1893]. The transformation of X^m into Y^m is given by:

$$\begin{aligned} Y^m &= Y_R^m + iY_I^m \\ &= \nu(X_R^m, X_I^m) + i\omega(X_R^m, X_I^m) \\ &= \xi(X_R^m + iX_I^m) = \xi(X^m) \end{aligned} \tag{1}$$

This transformation showcases the method by which complex signal representations, governed by their underlying stochastic properties, are processed for predictive tasks. The true function, $\xi(\cdot)$, which comprises the real-valued functions, $\nu(\cdot)$ and $\omega(\cdot)$, acts on random variables X_R^m and X_I^m to yield Y_R^m and Y_I^m. Analytic signals are an extension of this framework and have no negative frequency components.

2.1.1 Analytic Signal Representation

The analytic signal representation is defined as $X_I^m = \mathcal{H}\{X_R^m\}$, where for $x_R^m \sim X_R^m$, we have $x_I^m = \mathcal{H}\{x_R^m\} \in \mathbb{R}^{dN}$, $\forall m \in \mathcal{V}$. This construction is formalized as:

$$X^m = X_R^m + i\,\mathcal{H}\{X_R^m\}, \quad \text{where:} \tag{2}$$

$$\mathcal{H}\{X_R^m\}[n] = \frac{2}{dN} \sum_{u \in \mathcal{U}'} X_R^m[u]\, \cot\Big[(u-n)\frac{\pi}{dN}\Big]. \tag{3}$$

$\mathcal{H}\{X_R^m\}$ is the discrete Hilbert transform (DHT) of X_R^m, where $\mathcal{U}' = \{\, u \in \{0, 1, \ldots, dN-1\}, u \not\equiv n \pmod 2 \,\}$, which selects even indices when n is odd and odd indices when n is even. The DHT introduces a phase shift of $-90°$ to all the positive frequency components of X_R^m

and $+90^\circ$ to all the negative frequency components of X_R^m, establishing orthogonality between X_R^m and X_I^m. We observe that $\mathcal{H}\{\cdot\}$ is bijective since it is invertible, wherein $-\mathcal{H}\{\mathcal{H}\{X_R^m\}\} = X_R^m$, $\forall m \in \mathcal{V}$.

2.2 Related Work

The development of neural networks for complex signal processing has led to extensive research comparing the effectiveness of complex-valued neural networks (CVNNs) versus real-valued neural networks (RVNNs). While CVNNs are theoretically capable of capturing the information contained in phase components — see radar imaging [Gao et al., 2019], electromagnetic inverse scattering [Guo et al., 2021], MRI fingerprinting [Virtue et al., 2017], and automatic speech recognition [Shafran et al., 2018] — practical implementation is yet to show considerable improvements over RVNNs. In particular, [Guberman, 2016] depicts that RVNNs tend to have significantly lower training losses compared to CVNNs, and that comparing test losses, RVNNs still marginally outperform CVNNs in terms of generalization despite their vulnerability to overfitting. When the overfitting is substantial [Barrachina et al., 2021], RVNNs, despite their simpler training and optimization, observe worse generalization than CVNNs. A similar result is illustrated in [Trabelsi et al., 2018], where multidimensional RVNNs, with concatenated real and imaginary components fed into individual channels, exhibit performance metrics closely aligned with those of CVNNs, especially in architectures with constrained parameter sizes. Furthermore, CVNNs introduce higher computational complexity and encounter challenges in formulating holomorphic activation functions [Lee et al., 2022]. These studies reveal a balance between the theoretical benefits of CVNNs and the practical efficiencies of RVNNs. While CVNNs have received tremendous attention in recent years, the optimization of RVNNs for complex signal processing remains an open problem, especially in regard to improved training and regularization techniques for enhanced generalization performance and latent space interpretability.

3 Steinmetz Neural Networks

Reflecting on the inherent challenges and benefits of both RVNNs and CVNNs, we target a framework that leverages the simplicity in training offered by RVNNs while offering improved generalization in the processing of complex signals. Within this context, multi-view representation fusion emerges as a potential framework, proposing that different perspectives — or 'views' — of data can provide complementary information, thereby enhancing learning and generalization [Sun, 2013, Xu et al., 2013, Lahat et al., 2015, Zhao et al., 2017, Yan et al., 2021]. Informed by this principle, we introduce the *Steinmetz Neural Network* architecture, which is designed to process the real (X_R^m) and the imaginary (X_I^m) parts of complex signal representations as separate views before joint processing. We formalize how this architecture leverages the complementarity principle of multi-view learning in Section 3.1.

3.1 Theoretical Foundations

In the context of neural networks' information-theoretic foundations, [Tishby and Zaslavsky, 2015] leveraged the Data Processing Inequality to characterize the architecture illustrated in Figure 1. This architecture aligns with the classical RVNN architecture from the literature on CVNNs [Trabelsi et al., 2018]. Here, $\xi(\cdot)$ denotes the true function, $h(\psi(\cdot))$ describes the neural network, and $\hat{Y}^m = h(\psi(X^m))$ denotes the predictions. Accordingly, we have that $I(Y^m; X^m) \geq I(Y^m; Z^m) \geq I(Y^m; \hat{Y}^m)$. Regarding practical implementation, the input space, $[[X_R^m]^T, [X_I^m]^T]^T \in \mathbb{R}^{2dN}$, is jointly processed by $\psi^*(\cdot)$, using individual channels for X_R^m and X_I^m to form the latent space, $[[Z_R^m]^T, [Z_I^m]^T]^T \in \mathbb{R}^{2lN}$, which then gets jointly processed by $h^*(\cdot)$, using individual channels for Z_R^m and Z_I^m, to obtain the output space, $\hat{Y}^m \in \mathbb{C}^k$.

Building upon this architecture, the Steinmetz neural network postulates the Markov chain depicted in Figure 2, where the random variables, $Z_I^m = f(X_I^m)$ and $Z_R^m = g(Z_R^m)$, are the respective outputs of the parallel subnetworks, $f(\cdot)$ and $g(\cdot)$, where $Z^m = Z_R^m + iZ_I^m$ denotes the latent representation, and $\hat{Y}^m = h(Z^m)$ denotes the predictions yielded by the shared network, $h(\cdot)$ [1]. Per the construction in Figure 2, the Steinmetz neural network is given by:

$$\begin{aligned} \hat{Y}^m &= h(Z_R^m + iZ_I^m) \\ &= h(g(X_R^m) + if(X_I^m)) = h(\psi(X^m)). \end{aligned} \tag{4}$$

This architecture exhibits several distinctions from the classical RVNN. For practical implementation, X_R^m and X_I^m are processed by two different neural networks to form Z_R^m and Z_I^m, respectively, which are concatenated in the latent space to form $[[Z_R^m]^T, [Z_I^m]^T]^T \in \mathbb{R}^{2lN}$, and jointly processed by $h^*(\cdot)$ using individual channels for Z_R^m and Z_I^m. The rationale behind this initial separate processing step stems from the complementarity principle of multi-view learning [Xu et al., 2013]. This principle suggests that before forming a shared latent space in a multi-view setting, separately processing views that contain unique information can improve the interpretability of representations. We now define relevant terms to extend this notion to our Steinmetz neural network architecture.

[1] For practical implementation, we mean center Z_R^m and Z_I^m as the final step before concatenation.

Figure 1: Classical RVNN Markov chain (left) and practical implementation (right)

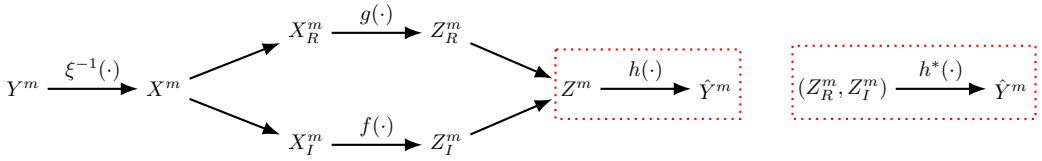


Figure 2: Steinmetz neural network Markov chain (left) and practical implementation (right)

Suppose $X_R^m = (Z_R^m, \Lambda_R^m)$, $Z_R^m \perp\!\!\!\perp \Lambda_R^m$, where Z_R^m is the latent representation from X_R^m that contains information relevant to Y^m when combined with Z_I^m, and Λ_R^m is information in X_R^m irrelevant to Y^m when combined with Z_I^m. Let $X_I^m = (Z_I^m, \Lambda_I^m)$, $Z_I^m \perp\!\!\!\perp \Lambda_I^m$, wherein Z_I^m is the latent representation from X_I^m that contains information relevant to Y^m when combined with Z_R^m, and Λ_I^m is information in X_I^m irrelevant to Y^m when combined with Z_I^m. Let $(Z_R^m, Z_I^m) = (\hat{Y}^m, \Gamma^m)$, $\hat{Y}^m \perp\!\!\!\perp \Gamma^m$, where (Z_R^m, Z_I^m) is a sufficient statistic of (X_R^m, X_I^m) with respect to Y^m. Here, $\hat{Y}^m$ is a minimal sufficient statistic of (Z_R^m, Z_I^m) with respect to Y^m, and Γ^m is information in (Z_R^m, Z_I^m) irrelevant to Y^m.

For the classical RVNN, we jointly process (X_R^m, X_I^m) and aim to form $(Z_R^m, Z_I^m) = \psi^*(X_R^m, X_I^m)$, filtering out $(\Lambda_R^m, \Lambda_I^m)$, where (Z_R^m, Z_I^m), is a sufficient statistic of (X_R^m, X_I^m) with respect to Y^m. We then jointly process (Z_R^m, Z_I^m) and attempt to form the predictions, $\hat{Y}^m = h^*(Z_R^m, Z_I^m)$, filtering out Γ^m. We refer to this approach as *joint-only processing*. For the Steinmetz neural network, we separately process X_R^m and X_I^m, and attempt to form $Z_R^m = g(X_R^m)$ and $Z_I^m = f(X_I^m)$, filtering out Λ_R^m and Λ_I^m, where (Z_R^m, Z_I^m), is a sufficient statistic of (X_R^m, X_I^m) with respect to Y^m. Paralleling the joint-only processing case, we jointly process this latent space, aiming to form $\hat{Y}^m = h^*(Z_R^m, Z_I^m)$. We refer to this approach as *separate-then-joint processing*.

In comparing the Steinmetz neural network's separate-then-joint processing scheme with the more classical joint-only processing scheme, we note that the former approach enables us to train individual subnetworks in place of $g(\cdot)$ and $f(\cdot)$ to filter out the respective information irrelevant to Y^m present within X_R^m and X_I^m. Contrarily, training a single neural network, $\psi^*(\cdot)$, via the joint-only processing scheme requires handling X_R^m and X_I^m simultaneously, meaning the network must learn to generalize across potentially disparate noise distributions and data characteristics. This makes it challenging to optimize the filtering of Λ_R^m and Λ_I^m if their properties differ significantly.

Accordingly, to characterize the complexity of representing (Z_R^m, Z_I^m) from (X_R^m, X_I^m) in the Steinmetz neural network and classical RVNN approaches, we propose the following construction. Let $\mathbf{\Sigma_J}$ denote the matrix of covariances of X_R^m, X_I^m posited by the joint-only processing approach, and let $\mathbf{\Sigma_S}$ denote the matrix of covariances of X_R^m, X_I^m posited by the separate-then-joint processing approach, where:

$$\mathbf{\Sigma_J} = \begin{bmatrix} \mathbf{K}_{X_R^m} & \mathbf{K}_{X_R^m, X_I^m} \\ \mathbf{K}_{X_I^m, X_R^m} & \mathbf{K}_{X_I^m} \end{bmatrix}, \tag{5}$$

$$\mathbf{\Sigma_S} = \begin{bmatrix} \mathbf{K}_{X_R^m} & \overline{\mathbf{K}}_{X_R^m, X_I^m} \\ \overline{\mathbf{K}}_{X_I^m, X_R^m} & \mathbf{K}_{X_I^m} \end{bmatrix}, \tag{6}$$

$$\mathbf{K}_{X_R^m, X_I^m} = \begin{bmatrix} \mathbf{K}_{Z_R^m, Z_I^m} & \mathbf{K}_{Z_R^m, \Lambda_I^m} \\ \mathbf{K}_{\Lambda_R^m, Z_I^m} & \mathbf{K}_{\Lambda_R^m, \Lambda_I^m} \end{bmatrix}, \tag{7}$$

$$\overline{\mathbf{K}}_{X_R^m, X_I^m} = \begin{bmatrix} \mathbf{0}_{kN \times kN} & \mathbf{K}_{Z_R^m, \Lambda_I^m} \\ \mathbf{K}_{\Lambda_R^m, Z_I^m} & \mathbf{K}_{\Lambda_R^m, \Lambda_I^m} \end{bmatrix}. \tag{8}$$

We note that $\mathbf{K}_{Z_R^m, Z_I^m} = \mathbf{0}_{dN \times dN}$ in the separate-then-joint processing approach, since $f(\cdot)$ and $g(\cdot)$ do not consider the interactions between Z_R^m and Z_I^m. This is characterized in Section B.1 of the Appendix. To measure the magnitude of the data interactions across both approaches, we consider the $L_{p,q}$ norm, with $p, q \geq 1$, of $\mathbf{\Sigma_J}$ and $\mathbf{\Sigma_S}$. As $\mathbf{\Sigma_J}$ includes the aforementioned cross-covariance matrix of Z_R^m and Z_I^m, it follows that $\|\mathbf{\Sigma_J}\|_{p,q} \geq \|\mathbf{\Sigma_S}\|_{p,q}$, as shown in Corollary 3.1.

Corollary 3.1 *Let $\mathbf{\Sigma_J}$ be the matrix of covariances of X_R^m, X_I^m from joint-only processing, and let $\mathbf{\Sigma_S}$ be the matrix of covariances of X_R^m, X_I^m from separate-then-joint processing. It follows that:*

$$\|\mathbf{\Sigma_J}\|_{p,q} \geq \|\mathbf{\Sigma_S}\|_{p,q}, \quad \textit{where:} \tag{9}$$

$$Z_R^m \perp\!\!\!\perp Z_I^m \implies \|\mathbf{\Sigma_J}\|_{p,q} = \|\mathbf{\Sigma_S}\|_{p,q}. \tag{10}$$

This greater norm indicates reduced interpretability, as the presence of cross-covariance terms implies that joint-only processing must not only handle Z_R^m and Z_I^m individually, but also their interactions, which can complicate the representation process. As such, $\mathbf{\Sigma_S}$ has

a smaller $L_{p,q}$ norm and associated representational complexity, enabling the Steinmetz network to separately extract Z_R^m and Z_I^m without the added burden of accounting for interactions. This Steinmetz architecture can also be leveraged to obtain a smaller upper bound on the generalization gap, as per Section 4.

4 Consistency Constraint

Suppose ψ_S, f_S, and g_S are stochastic transformations, where $\theta_{\psi_S} = (\theta_{f_S}, \theta_{g_S})$ is a random variable denoting the parameters of ψ_S, with the variables, θ_{f_S} and θ_{g_S}, parameterizing f_S and g_S, respectively. Per [Federici et al., 2020, Fischer, 2020, Lee et al., 2021], obtaining an optimal latent representation, Z^m, can be formulated as minimizing the mutual information between X^m and Z^m, conditioned on Y^m. However, as explored by [Hafez-Kolahi et al., 2020], this framework does not hold when the encoder, ψ_s, is learned with the training dataset, s. To avoid this counterexample, [Kawaguchi et al., 2023] proposed an additional term that captures the amount of information in S that is used to train the encoder, ψ_S. As such, in the context of our Steinmetz neural network architecture, obtaining the optimal Z^m can be found by minimizing the expression in Eq. (11), where $\theta_{\psi_S} \in \mathbb{R}^c, \theta_{f_S} \in \mathbb{R}^{c_1}, \theta_{g_S} \in \mathbb{R}^{c_2}$.

$$\begin{aligned}\mathcal{J}(Z^m) &= I(X^m; Z^m|Y^m) + I(S; \theta_{\psi_S}) \\ &= I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S})\end{aligned} \tag{11}$$

Per Eq (11), the optimal latent representation, Z^m, best captures relevant information from X^m about Y^m while also considering the influence of S on the encoder parameters, θ_{ψ_S}. We now consider the upper bound on the generalization gap over the training dataset, Δs, which is an adapted version of the bound originally proposed in [Kawaguchi et al., 2023].

Theorem 4.1 *For any $\delta > 0$ with probability at least $1 - \delta$ over the training dataset, s, we have that:*

$$\Delta s = \left[\mathbb{E}\big[\ell(\hat{Y}^m, Y^m)\big] - \frac{1}{M}\sum_{m\in\mathcal{V}} \ell(\hat{y}^m, y^m)\right] \leq K(Z^m),$$
$$\text{where: } K(Z^m) = \frac{K_1(\alpha)}{\sqrt{M}} + K_2\sqrt{\frac{\mathcal{J}(Z^m)\log(2) + K_3}{M}}. \tag{12}$$

The complete formulas for $K_1(\alpha)$, K_2, and K_3 can be found in Section A of the Appendix. We note that $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ is a bounded per-sample loss function.

The upper bound from Eq. (12) captures the tradeoff between how well the latent space encapsulates information about the labels, and how much the encoder overfits the training distribution, wherein smaller values of $[I(X^m; Z^m|Y^m) + I(S; \theta_{\psi_S})]$ yield a smaller upper bound on the generalization gap. Accordingly, we pose the following inquiry: *is it possible to leverage the Steinmetz neural network architecture to obtain a smaller upper bound on the generalization gap*? To this end, we establish the existence of a lower bound, $\mathcal{D}(Z^m)$, on $[I(X^m; Z^m|Y^m) + I(S; \theta_{\psi_S})]$ that is achievable using a constraint on the latent space of the Steinmetz neural network. We formalize this in Corollary 4.2.

Corollary 4.2 *Suppose $\mathcal{F}^m$ denotes the set of all constraints on Z^m. We have that $\forall m \in \mathcal{V}$:*

$$\mathcal{D}(Z^m) \leq \mathcal{J}(Z^m),\ \forall Z^m \in \mathbb{C}^{lN} \tag{13}$$
$$\exists f \in \mathcal{F}^m\ :\ \forall Z^m \in \mathcal{E},\ \mathcal{D}(Z^m) = \mathcal{J}(Z^m). \tag{14}$$

Where $\mathcal{E} = \{Z^m \in \mathbb{C}^{lN} | f\}$ denotes the set of all Z^m satisfying the constraint $f \in \mathcal{F}^m$.

Achieving a smaller upper bound on the generalization gap is indicative of a network's potential for improved accuracy in making predictions on unseen data. This relationship is deeply rooted in the notions of statistical learning theory, and more formally, in Structural Risk Minimization (SRM) and VC theory [Vapnik and Chervonenkis, 1971, Vapnik, 1999]. Consequently, should there exist a consistency constraint on the latent space yielding a smaller upper bound on Δs, we would expect it to improve the Steinmetz neural network's capacity to generalize [Vapnik, 2013].

We have proven Corollary 4.2 in Section A of the Appendix, and present the lower bound, $\mathcal{D}(Z^m)$, in Theorem 4.3, wherein there exists a consistency constraint ensuring the achievability of $\mathcal{D}(Z^m)$.

Theorem 4.3 *Consider $X^m = X_R^m + iX_I^m \in \mathbb{C}^{dN}$, $Y^m \in \mathbb{C}^k$, and $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$, with $\theta_{\psi_S} \in \mathbb{R}^c, \theta_{g_S} \in \mathbb{R}^{c_2}$. It follows that $\mathcal{D}(Z^m) \leq \mathcal{J}(Z^m)$, where:*

$$\begin{aligned}\mathcal{D}(Z^m) = H(Z_R^m) - I(Y^m; Z^m) - M\big[&H(X^m|\theta_{\psi_S}) \\ &- H(Y^m|X_R^m + iX_I^m, \theta_{g_S}) - H(Y^m) \\ &- H(X_R^m|Y^m) - H(iX_I^m|X_R^m, iZ_I^m, Y^m)\big].\end{aligned} \tag{15}$$

With equality if the following condition holds $\forall m \in \mathcal{V}$:

$$\begin{aligned}&\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN}\ : \\ &Z_I^m = \phi(Z_R^m) \implies \mathcal{J}(Z^m) = \mathcal{D}(Z^m).\end{aligned} \tag{16}$$

Theorem 4.3 informs us that $\mathcal{D}(Z^m)$ is achievable when we enforce $Z_I^m = \phi(Z_R^m)$, where $\phi(\cdot)$ is a deterministic, bijective function. We note that as the Steinmetz neural network is trained to minimize the average loss on the training dataset (through empirical risk minimization), we expect Z^m to become more informative about the labels, whereby $I(Y^m; Z^m)$ increases. We now further extend this result to Theorem 4.1, via which we obtain a smaller upper bound on the generalization gap.

Theorem 4.4 *For any $\delta > 0$ with probability at least $1 - \delta$ over the training dataset, s, we have that:*

$$\Delta s \leq G(Z^m) \leq K(Z^m),$$
$$\text{where: } G(Z^m) = \frac{K_1(\alpha)}{\sqrt{M}} + K_2\sqrt{\frac{\mathcal{D}(Z^m)\log(2) + K_3}{M}}. \tag{17}$$

With $G(Z^m) = K(Z^m)$ if the following condition holds:

$$\begin{aligned} &\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN} : \\ &Z_I^m = \phi(Z_R^m) \implies G(Z^m) = K(Z^m). \end{aligned} \tag{18}$$

The complete formulas for $K_1(\alpha)$, K_2, and K_3 can be found in Section A of the Appendix.

5 Analytic Neural Network

As detailed in Section 4, by introducing a constraint within the latent representation, Z^m, such that Z_R^m and Z_I^m, are related by a deterministic, bijective function, $\phi(\cdot)$, we can leverage improved control over the generalization gap, Δs. A natural question that follows is: *which $\phi(\cdot)$ should be chosen to improve predictive performance*? To address this, we consider a configuration that focuses on the properties and predictive advantages of orthogonal latent representations.

In feature engineering literature, selecting features that are orthogonal to others is a common strategy to minimize redundancy and improve network performance [Chaudhry et al., 2020]. For our application, with Z_R^m and Z_I^m as the latent features, we aim to use a function, $\phi(\cdot)$, which ensures these features are as orthogonal as possible. This objective aligns with the above principle of using non-redundant features. We revisit the analytic signal construction from Section 2.1.1, wherein the real and imaginary parts are related by the DHT, and are orthogonal to each other. Accordingly, for $\phi(\cdot) = \mathcal{H}\{\cdot\}$, where $Z^m = Z_R^m + iZ_I^m$, with $Z_I^m = \mathcal{H}\{Z_R^m\}$, it follows that Z_R^m and Z_I^m are orthogonal (see Section B.2 of the Appendix). We formalize this in Corollary 5.1.

Corollary 5.1 *Consider $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$, $\langle \cdot, \cdot \rangle : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}_{\geq 0}$. We have that $\forall m \in \mathcal{V}$:*

$$Z_I^m = \mathcal{H}\{Z_R^m\} \implies \langle Z_R^m, Z_I^m \rangle = 0. \tag{19}$$

We term this Steinmetz neural network, where $Z_I^m \to \mathcal{H}\{Z_R^m\}$ during training as the *Analytic Neural Network*. In leveraging $\phi(\cdot) = \mathcal{H}(\cdot)$ as our consistency constraint, we provide a framework that encourages orthogonality between Z_R^m and Z_I^m, aiming to improve the Steinmetz neural network's predictive capabilities. We outline the practical implementation of this consistency constraint through the *Hilbert Consistency Penalty*.

Consider sets $\{x_R^m,\ m \in \mathcal{V}\}$ and $\{x_I^m,\ m \in \mathcal{V}\}$ from the training dataset, with $x_R^m, x_I^m \in \mathbb{R}^{dN}$. It follows that $z_R^m = g(x_R^m)$ and $z_I^m = f(x_I^m)$, wherein $z_R^m, z_I^m \in \mathbb{R}^{lN}$. To implement the Hilbert consistency penalty, we make use of the discrete Fourier transform (DFT), leveraging its properties in relation to phase shifts — we consider $F_R^m = \mathcal{F}\{z_R^m\} \in \mathbb{C}^{lN}$ as the DFT of z_R^m where b denotes the frequency index. Eq. (20) summarizes the frequency domain implementation of this phase shift.

$$H_R^m[b] = \begin{cases} F_R^m[b] \cdot (-i) & \text{for } 0 < b < \frac{lN}{2} \\ F_R^m[b] & \text{for } b = 0,\ \frac{lN}{2} \\ F_R^m[b] \cdot (i) & \text{for } \frac{lN}{2} < b < lN \end{cases}, \tag{20}$$

$$\text{where: } F_R^m[b] = \sum_{n=0}^{lN-1} z_R^m[n] e^{-\frac{i2\pi bn}{lN}}. \tag{21}$$

Above, $H_R^m \in \mathbb{C}^{lN}$ denotes the frequency components of $\mathcal{H}\{z_R^m\}$. Applying the inverse FFT to H_R^m yields the discrete Hilbert transform of z_R^m, $\mathcal{H}\{z_R^m\}$, in the time domain, as detailed in Eq. (22).

$$\begin{aligned} \mathcal{H}\{z_R^m\}[n] &= \mathcal{F}^{-1}\{H_R^m\}[n] \\ &= \frac{1}{lN} \sum_{b=0}^{lN-1} H_R^m[b] e^{\frac{i2\pi bn}{lN}}. \end{aligned} \tag{22}$$

We implement the Hilbert consistency penalty by penalizing the average error between $\mathcal{H}\{z_R^m\}$ and z_I^m, which we denote as $\mathcal{L}_\mathcal{H}$, where $\ell_\mathcal{H}$ is the relevant error metric. This penalty summarized in Definition 5.2.

Definition 5.2 *Consider $z_R^m, z_I^m \in \mathbb{R}^{lN}$, and suppose $H_R^m = \mathcal{F}\{\mathcal{H}\{z_R^m\}\} \in \mathbb{C}^{lN}$. We have that:*

$$\mathcal{L}_\mathcal{H} = \frac{1}{M} \sum_{m=0}^{M-1} \ell_\mathcal{H}\left(\mathcal{H}\{z_R^m\}, z_I^m\right), \tag{23}$$

$$\text{where: } \mathcal{L}_\mathcal{H} = 0 \iff z_I^m = \mathcal{H}\{z_R^m\},\ \forall m \in \mathcal{V}. \tag{24}$$

We note that $\ell_\mathcal{H} : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}_{\geq 0}$ is a bounded per-sample loss function.

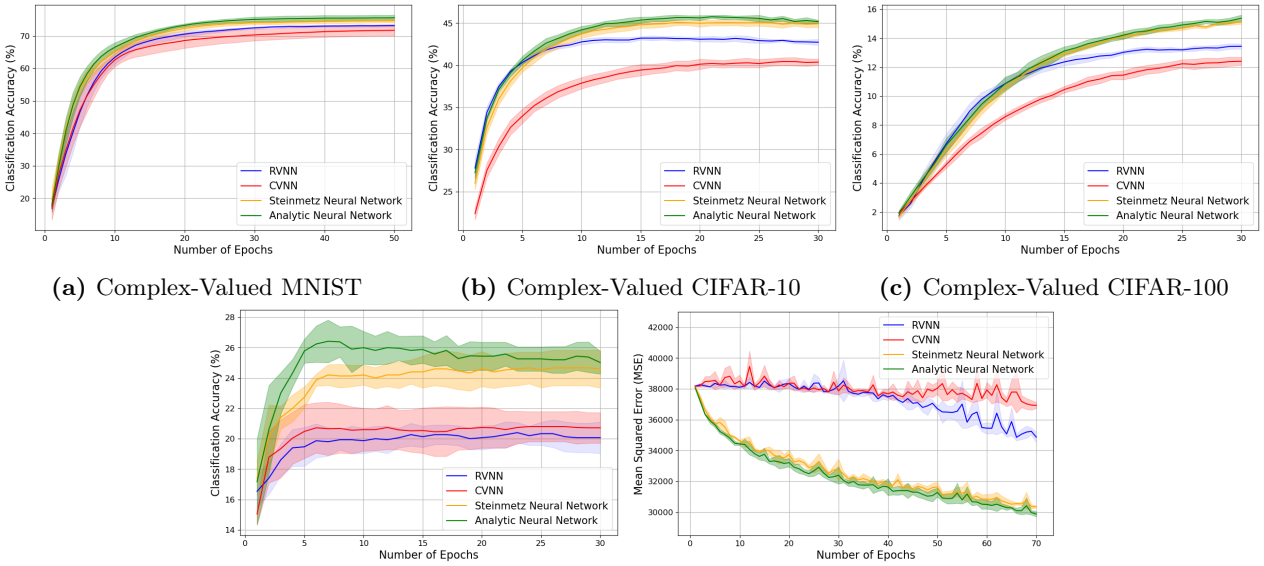
The cumulative loss function used to train the analytic neural network is derived as the weighted sum of the average loss on the training dataset and the Hilbert consistency penalty, where β is the tradeoff parameter. During training, we jointly minimize the error on the training dataset, and encourage the network to form analytic signal representations in the latent space. The overall loss, $\mathcal{L}$, is given by:

$$\mathcal{L} = \frac{1}{M} \sum_{m=0}^{M-1} \ell\left(\hat{y}^m, y^m\right) + \beta \mathcal{L}_\mathcal{H}. \tag{25}$$

Where the value of β can be fine-tuned to optimize the predictive accuracy on the training dataset, s.

Table 1: Test performance comparison on CV-MNIST ($M = 500$), CV-CIFAR-10 ($M = 50{,}000$), CV-CIFAR-100 ($M = 50{,}000$), CV-FSDD ($M = 2{,}700$), and RASPNet ($M = 20{,}000$).

Dataset	`RVNN` **Test Perf.**	`CVNN` **Test Perf.**	`Steinmetz` **Test Perf.**	`Analytic` **Test Perf.**
CV-MNIST	73.180±0.407 (%)	71.716±1.957 (%)	74.680±0.722 (%)	75.580±0.970 (%)
CV-CIFAR-10	42.734±0.397 (%)	40.370±0.417 (%)	44.922±0.474 (%)	45.180±0.239 (%)
CV-CIFAR-100	13.444±0.197 (%)	12.412±0.319 (%)	15.110±0.137 (%)	15.380±0.256 (%)
CV-FSDD	20.067±1.036 (%)	20.720±1.006 (%)	24.600±1.259 (%)	25.020±0.749 (%)
RASPNet	34867±347.8 (MSE)	36928±197.9 (MSE)	30360±70.12 (MSE)	29880±169.9 (MSE)
Dataset	**Parameters**	**Parameters**	**Parameters**	**Parameters**
CV-MNIST	52,970	55,764	53,002	53,002
CV-CIFAR-10	199,402	202,196	199,434	199,434
CV-CIFAR-100	205,252	213,896	205,284	205,284
CV-FSDD	1,622,310	1,644,620	1,622,410	1,622,410
RASPNet	223,682	232,324	223,746	223,746



(a) Complex-Valued MNIST **(b)** Complex-Valued CIFAR-10 **(c)** Complex-Valued CIFAR-100

(d) Complex-Valued FSDD **(e)** RASPNet Scenario $i = 29$

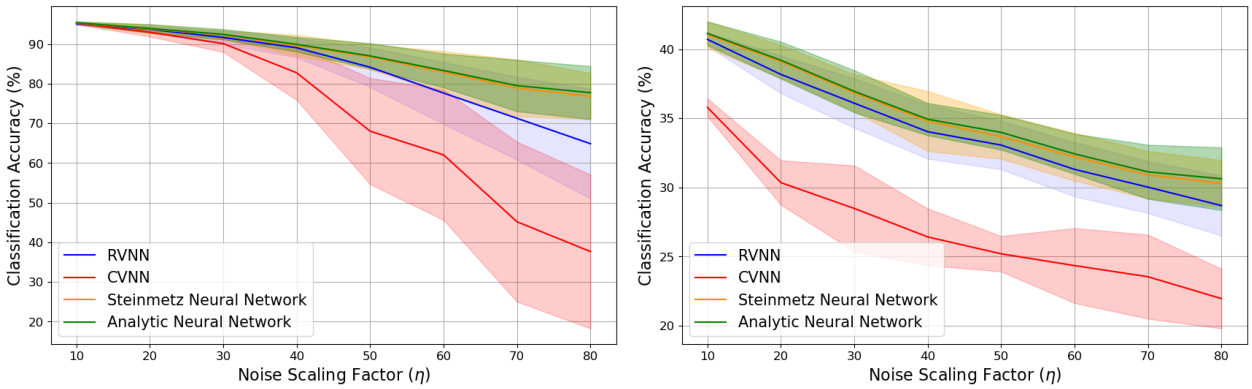
Figure 3: Test performance comparison on CV-MNIST ($M = 500$), CV-CIFAR-10 ($M = 50{,}000$), CV-CIFAR-100 ($M = 50{,}000$), CV-FSDD ($M = 2{,}700$), and RASPNet ($M = 20{,}000$) through CVNN, RVNN, Steinmetz neural network, and analytic neural network. The x-axis indicates the training epochs, while the y-axis indicates the test performance (classification accuracy and mean squared error).

6 Empirical Results

We present empirical results on benchmark datasets for complex-valued multi-class classification and regression, and on a synthetic signal processing example for complex-valued regression. Per [Trabelsi et al., 2018] and [Scardapane et al., 2020], we present classification results on complex-valued MNIST [Deng, 2012], CIFAR-10 [Krizhevsky et al., 2009], CIFAR-100 [Krizhevsky et al., 2009], and FSDD [Jackson et al., 2018]. We also present a regression result on the RASPNet benchmark dataset [Venkatasubramanian et al., 2024], which comprises intrinsically complex-valued radar returns. The CVNNs in this analysis were constructed through the Complex Pytorch library [Matthès et al., 2021], which implements the layers proposed in [Trabelsi et al., 2018].

6.1 Benchmark Datasets

The first experiment we consider is an assessment of our methods on Complex-Valued MNIST (CV-MNIST), Complex-Valued CIFAR-10 (CV-CIFAR-10), Complex-Valued CIFAR-100 (CV-CIFAR-100), and the audio-based Complex-Valued Free Spoken Digit Dataset (CV-



(a) Noisy Complex-Valued MNIST

(b) Noisy Complex-Valued CIFAR-10

Figure 4: Noise robustness test performance on CV-MNIST ($M = 60{,}000$) and CV-CIFAR-10 ($M = 50{,}000$) through CVNN, RVNN, Steinmetz neural network, and analytic neural network. The x-axis is the scaling factor, η, for the additive complex normal noise, while the y-axis indicates the classification accuracy.

Table 2: Test MSE (magnitude and phase) for channel identification task with $\rho = \sqrt{2}/2$.

	RVNN	CVNN	Steinmetz	Analytic
Magnitude Error	1.035$\pm$0.011	1.003$\pm$0.006	1.151$\pm$0.005	1.114$\pm$0.078
Phase Error	4.270$\pm$0.095	4.451$\pm$0.085	3.808$\pm$0.129	3.768$\pm$0.169
Parameters	2,594	4,738	2,626	2,626

FSDD), for multi-class classification, and on RASPNet for regression, evaluating the efficacy of the proposed Steinmetz and analytic neural networks when there are no ablations introduced within the data. We obtain CV-MNIST by taking a $dN = 784$-point DFT of each of the $M = 500$ training images (a small subset of the first 500 images in MNIST). The test set is formed by taking a 784-point DFT of each of the 10,000 test images. We obtain CV-CIFAR-10 and CV-CIFAR-100 by taking a $dN = 3072$-point DFT of each of the $M = 50{,}000$ training images. The test set is constructed by taking a 3072-point DFT of each of the 10,000 test images. We obtain CV-FSDD by taking a $dN = 8000$-point DFT of each of the $M = 2{,}700$ training audio signals. The test set is constructed by taking a 8000-point DFT of each of the 300 test audio signals. For CV-MNIST, CV-CIFAR-10, and CV-FSDD we have $k = 10$, and for CV-CIFAR-100, we have $k = 100$. RASPNet consists of $M = 20{,}000$ training and 5,000 test radar returns of size $dN = 1680$, with $k = 2$ (2D target localization).

We train a RVNN, CVNN, Steinmetz neural network, and analytic neural network using Cross Entropy Loss for $lN = 64$ to classify the images in CV-MNIST, CV-CIFAR-10, and CV-CIFAR-100, $lN = 200$ to classify the audio signals in CV-FSDD, and using MSE Loss for $lN = 128$ to localize targets from the radar returns in RASPNet. These neural network architectures and hyperparameter choices are described in Section D of the Appendix, wherein we leverage the Adam optimizer [Kingma and Ba, 2014] to train each architecture using a fixed learning rate. The empirical results pertaining to this first experiment are depicted in Table 1 and Figure 3. On CV-MNIST, CV-CIFAR-10, CV-CIFAR-100, CV-FSDD, and RASPNet, we observe that the Steinmetz and analytic neural networks achieve improved generalization over the classical RVNN and the CVNN. The analytic neural network achieves the highest classification accuracy and the lowest MSE.

The second experiment is an examination of the impact of additive complex normal noise on the performance of our proposed neural networks on CV-MNIST and CV-CIFAR-10. We add standard complex normal noise scaled by a factor, η, to each example $x^m \in s$, where $M = 60{,}000$ for CV-MNIST and $M = 50{,}000$ for CV-CIFAR-10. The new training dataset, s', is given by:

$$s' = \{(x^{m'}, y^m), m \in \mathcal{V}\}, \tag{26}$$

$$\text{where: } x^{m'} = x^m + \eta \times \mathcal{CN}(\mathbf{0}, \mathbf{I}_{dN}). \tag{27}$$

This experimental setup allows us to gauge how the signal-to-noise ratio (SNR) influences the efficacy of our Steinmetz and analytic neural networks. The empirical results for this experiment are given in Figure 4. We see that across both datasets, the Steinmetz and analytic neural networks are far more resilient to additive noise.

6.2 Channel Identification

We evaluate our proposed Steinmetz and analytic networks on the benchmark channel identification task from [Scardapane et al., 2020, Bouboulis et al., 2015]. Let $X^m = \sqrt{1-\rho^2}\bar{X}^m + i\rho\tilde{X}^m$ denote the input to the channel, wherein $\bar{X}^m$ and $\tilde{X}^m$ are Gaussian random variables, and ρ determines the circularity of the signal. Here, $dN = 5$ denotes the length of the input sequence, an embedding of the channel inputs over dN time steps. The channel output, $Y^m \in \mathbb{C}^k$, is formed using a linear filter, a memoryless nonlinearity, and by adding white Gaussian noise to achieve an SNR of 5 dB, wherein X^m and Y^m are equivalent to s_n and r_n from [Scardapane et al., 2020]. We consider $M = 1000$ training examples and 1000 test examples. Each of the real-valued architectures output a $2k = 2$D vector, $[[\hat{Y}_R^m]^T, [\hat{Y}_I^m]^T]^T$, where $\hat{Y}_R^m, \hat{Y}_I^m \in \mathbb{R}$, and the CVNN outputs a $k = 1$-dimensional scalar, $\hat{Y}^m = \hat{Y}_R^m + i\hat{Y}_I^m$, which we reshape to form $[[\hat{Y}_R^m]^T, [\hat{Y}_I^m]^T]^T$. We train our Steinmetz and analytic neural networks, RVNN, and CVNN using MSE Loss, minimizing the distance between $[[\hat{Y}_R^m]^T, [\hat{Y}_I^m]^T]^T$ and $[[Y_R^m]^T, [Y_I^m]^T]^T$. We select $lN = 64$, and use the Adam optimizer to train each architecture using a fixed learning rate. We compute and report the MSE between the predicted and true magnitudes and phases on the test dataset in Table 2. From Table 2, we see while the magnitude prediction error is comparable between the CVNN, RVNN, and the Steinmetz and analytic neural networks, the latter pair observes a much lower phase prediction error.

7 Conclusion

In this work, we introduced Steinmetz neural networks, a new approach to processing complex-valued data using DNNs with parallel real-valued subnetworks. We provided its mathematical framework and outlined a consistency constraint to lower its generalization gap upper bound, and we presented the analytic neural network, which incorporates the consistency penalty, for practical implementation. We evaluated these networks on regression and classification benchmarks, showing improvements over existing RVNNs and CVNNs. Future work includes investigating more effective training techniques for Steinmetz neural networks and theoretical performance guarantees.

Acknowledgements

This work was supported in part by the U.S. Air Force Office of Scientific Research (AFOSR) under award FA9550-21-1-0235. Any opinions, findings and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Department of Defense.

References

J. A. Barrachina, C. Ren, C. Morisseau, G. Vieillard, and J.-P. Ovarlez. Complex-valued vs. real-valued neural networks for classification perspectives: An example on non-circular data. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2990–2994, 2021. doi: 10.1109/ICASSP39728.2021.9413814.

Joshua Bassey, Lijun Qian, and Xianfang Li. A survey of complex-valued neural networks, 2021.

Pantelis Bouboulis, Sergios Theodoridis, Charalampos Mavroforakis, and Leoni Evaggelatou-Dalla. Complex support vector machines for regression and quaternary classification. *IEEE Transactions on Neural Networks and Learning Systems*, 26(6):1260–1274, 2015. doi: 10.1109/TNNLS.2014.2336679.

Arslan Chaudhry, Naeemullah Khan, Puneet Dokania, and Philip Torr. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 33:9900–9911, 2020.

V. Cizek. Discrete hilbert transform. *IEEE Transactions on Audio and Electroacoustics*, 18(4):340–343, 1970. doi: 10.1109/TAU.1970.1162139.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *International Conference on Learning Representations*, 2020.

Ian Fischer. The conditional entropy bottleneck. *Entropy*, 22(9), 2020. ISSN 1099-4300. doi: 10.3390/e22090999.

Jingkun Gao, Bin Deng, Yuliang Qin, Hongqiang Wang, and Xiang Li. Enhanced radar imaging using a complex-valued convolutional neural network. *IEEE Geoscience and Remote Sensing Letters*, 16(1):35–39, 2019. doi: 10.1109/LGRS.2018.2866567.

Nitzan Guberman. On complex valued convolutional neural networks. *arXiv preprint arXiv:1602.09046*, 2016.

Liang Guo, Guanfeng Song, and Hongsheng Wu. Complex-valued pix2pix—deep neural network for nonlinear electromagnetic inverse scattering. *Electronics*, 10(6), 2021. ISSN 2079-9292. doi: 10.3390/electronics10060752.

Hassan Hafez-Kolahi, Shohreh Kasaei, and Mahdiyeh Soleymani-Baghshah. Sample complexity of classification with compressed input. *Neurocomputing*, 415:286–294, 2020. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2020.07.043.

Akira Hirose. Complex-valued neural networks. *Springer Science & Business Media*, 2012.

Zohar Jackson, César Souza, Jason Flaks, Yuxin Pan, Hereman Nicolas, and Adhish Thite. Jakobovski/free-spoken-digit-dataset: v1.0.8 (version v1.0.8)., 2018.

Kenji Kawaguchi, Zhun Deng, Xu Ji, and Jiaoyang Huang. How does information bottleneck help deep learning? In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 16049–16096. PMLR, 23–29 Jul 2023.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Dana Lahat, Tülay Adalı, and Christian Jutten. Multimodal data fusion: An overview of methods, challenges, and prospects. *Proceedings of the IEEE*, 103 (9):1449–1477, 2015.

ChiYan Lee, Hideyuki Hasegawa, and Shangce Gao. Complex-valued neural networks: A comprehensive survey. *IEEE/CAA Journal of Automatica Sinica*, 9 (8):1406–1426, 2022. doi: 10.1109/JAS.2022.105743.

Kuang-Huei Lee, Anurag Arnab, Sergio Guadarrama, John Canny, and Ian Fischer. Compressive visual representations. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

Maxime W. Matthès, Yaron Bromberg, Julien de Rosny, and Sébastien M. Popoff. Learning and avoiding disorder in multimode fibers. *Phys. Rev. X*, 11:021060, Jun 2021. doi: 10.1103/PhysRevX.11.021060. URL `https://link.aps.org/doi/10.1103/PhysRevX.11.021060`.

Simone Scardapane, Steven Van Vaerenbergh, Amir Hussain, and Aurelio Uncini. Complex-valued neural networks with nonparametric activation functions. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(2):140–150, 2020. doi: 10.1109/TETCI.2018.2872600.

Izhak Shafran, Tom Bagby, and R. J. Skerry-Ryan. Complex evolution recurrent neural networks (cernns). In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*, pages 5854–5858. IEEE, 2018. ISBN 978-1-5386-4658-8. doi: 10.1109/ICASSP.2018.8462556.

Josiah W. Smith. Complex-valued neural networks for data-driven signal processing and signal understanding, 2023.

Charles Proteus Steinmetz. *Theory and Calculation of Alternating Current Phenomena*. American Institute of Electrical Engineers, 1893.

Shiliang Sun. A survey of multi-view machine learning. *Neural Computing and Applications*, 23(7-8):2031–2038, 2013.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop (ITW)*. IEEE, 2015.

Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *International Conference on Learning Representations*, 2018.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10 (5):988–999, 1999.

Shyam Venkatasubramanian, Bosung Kang, Ali Pezeshki, Muralidhar Rangaswamy, and Vahid Tarokh. Raspnet: A benchmark dataset for radar adaptive signal processing applications. *arXiv preprint arXiv:2406.09638*, 2024.

Patrick Virtue, Stella X. Yu, and Michael Lustig. Better than real: Complex-valued neural nets for mri fingerprinting. *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3953–3957, 2017.

Jin-Hui Wu, Shao-Qun Zhang, Yuan Jiang, and Zhi-Hua Zhou. Complex-valued neurons can learn more but slower than real-valued neurons via gradient descent. pages 23714–23747, 2023.

Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

Xiaoqiang Yan, Shizhe Hu, Yiqiao Mao, Yangdong Ye, and Hui Yu. Deep multi-view learning methods: A review. *Neurocomputing*, 448:106–129, 2021. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2021.03.090.

Jing Zhao, Xing Xie, Xin Xu, and Shiliang Sun. Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54, 2017.

Appendix

A Consistency Constraint Derivation

As outlined in Section 4, our aim is to exploit the Steinmetz neural network architecture by deriving a consistency constraint that allows for improved control over the generalization gap, Δs. Recall Eq. 12, which provides an upper bound on Δs in terms of the mutual information between X^m and Z^m, between Y^m and Z^m, and between S and θ_{ψ_S}. The complete derivation of this bound is provided in [Kawaguchi et al., 2023].

Lemma A.1 *For any $\lambda > 0$, $\gamma > 0$ and $\delta > 0$ with probability at least $1 - \delta$ over s, we have that:*

$$\Delta s \leq K(Z^m) = \frac{K_1(\alpha)}{\sqrt{M}} + K_2 \sqrt{\frac{[I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S})] \log(2) + K_3}{M}}. \tag{28}$$

Where $\alpha = (I(\theta_{\psi_S}; S) + K_4) \log(2) + \log(2)$, and:

$$K_1(\alpha) = \frac{\max\limits_{m \in \mathcal{V}} \ell(\hat{y}^m, y^m) \sqrt{2\gamma |\mathcal{Y}^m|}}{M^{1/4}} \sqrt{\alpha + \log(2|\mathcal{Y}|/\delta)} + \gamma K_5, \tag{29}$$

$$K_2 = \max_{y^m \in \mathcal{Y}} \sum_{k=1}^{M_{y^m}} \ell((\mathfrak{z}_k^{y^m})^m, y^m) \sqrt{2|\mathcal{Y}| \mathbb{P}(Z^m = (\mathfrak{z}_k^{y^m})^m | Y = y^m)}, \tag{30}$$

$$K_3 = \left(\mathbb{E}_{y^m}[c_{y^m}(\theta_{\psi_s})] \sqrt{\left[p \log(\sqrt{M/\gamma}) \right] / 2} + K_4 \right) \log(2). \tag{31}$$

Above, for K_2, M_{y^m} denotes the size of the typical subset of the set of latent variables per $y^m \in \mathcal{Y}$, wherein the elements of the typical subset are given by: $\{(\mathfrak{z}_1^{y^m})^m, ..., (\mathfrak{z}_{M_{y^m}}^{y^m})^m\}$. For K_3, $c_{y^m}(\theta_{\psi_s})$ denotes the sensitivity of θ_{ψ_s}, $\theta_{\psi_s} \in \mathbb{R}^c$, and $K_4 = \frac{1}{\lambda} \log\left(\frac{1}{\delta e^{\lambda H(\theta_{\psi_S})}} \sum_{q \in \mathcal{M}} (\mathbb{P}(\theta_{\psi_S} = q))^{1-\lambda} \right) + H(\theta_{\psi_S} | S)$. Additionally, for K_1, we have that $K_5 = \max_{(x^m, y^m) \in (\mathcal{X} \times \mathcal{Y})} \ell(h(\psi_s(x^m)), y^m)$.

For improved control over the generalization gap, we form a smaller upper bound on Δs by deriving a lower bound, $\mathcal{D}(Z^m)$, on $[I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S})]$. We previously stated $\mathcal{D}(Z^m)$ is achievable by imposing a constraint on the latent representation, Z^m (see Corollary 4.2).

We now prove Corollary 4.2 and derive $\mathcal{D}(Z^m)$. We consider the expansion of the term $I(X^m; Z^m)$:

$$\begin{aligned} I(X^m; Z^m) &= H(Z^m) - H(Z^m | X^m) \\ &= H(Z_R^m + iZ_I^m) - H(g(X_R^m) + if(X_I^m) | X_R^m + iX_I^m) \\ &= H(Z_R^m) + H(iZ_I^m | Z_R^m) - H(Z_R^m | Z_R^m + iZ_I^m) \\ &= H(Z_R^m) + H(iZ_I^m | Z_R^m) \\ &\geq H(Z_R^m). \end{aligned} \tag{32}$$

This lower bound is achievable when there exists a deterministic function, $\phi(\cdot)$, relating Z_R^m and Z_I^m, wherein $H(iZ_I^m | Z_R^m) = 0$. We formalize this in Lemma A.2, where $\phi(\cdot)$ is bijective.

Lemma A.2 *Consider $X^m = X_R^m + iX_I^m \in \mathbb{C}^{dN}$, and $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$. Subsequently, it follows that $I(X^m; Z^m) \geq H(Z_R^m)$, with equality if the following condition holds $\forall m \in \mathcal{V}$:*

$$\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN} : Z_I^m = \phi(Z_R^m) \implies I(X^m; Z^m) = H(Z_R^m). \tag{33}$$

We now consider the expansion of $I(S;\theta_{\psi_S}) = H(S) - H(S|\theta_{\psi_S})$, and focus on the $H(S)$ term:

$$\begin{aligned} H(S) &= H\Big((X^0,Y^0),(X^1,Y^1),\ldots,(X^{M-1},Y^{M-1})\Big) \\ &= \sum_{m=0}^{M-1} H\Big((X^m,Y^m)\Big|(X^{m-1},Y^{m-1}),\ldots,(X^0,Y^0)\Big) \\ &= \sum_{m=0}^{M-1} H(X^m,Y^m) = M\big[H(X^m,Y^m)\big]. \end{aligned} \tag{34}$$

We note Eq. (34) follows from Section 2, since $S \sim P^{\otimes M}$. Thus, $H(X^i,Y^i|X^j,Y^j) = H(X^i,Y^i)$, $\forall i,j \in \mathcal{V}$. Expanding $H(X^m,Y^m)$, we have that:

$$\begin{aligned} H(S) &= M\big[H(X^m,Y^m,Z^m) - H(Z^m|X^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(Z^m|Y^m) + H(X^m|Z^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(Z^m|Y^m) + H(X_R^m + iX_I^m|Z^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(Z^m|Y^m) + H(X_R^m|Z^m,Y^m) + H(X_R^m + iX_I^m|X_R^m,Z^m,Y^m) \\ &\quad - H(X_R^m|X_R^m + iX_I^m,Z^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(Z^m,X_R^m|Y^m) + H(iX_I^m|X_R^m,Z^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(X_R^m|Y^m) + H(Z_R^m + iZ_I^m|X_R^m,Y^m) \\ &\quad + H(iX_I^m|X_R^m,Z_R^m + iZ_I^m,Y^m)\big] \\ &= M\big[H(Y^m) + H(X_R^m|Y^m) + H(iZ_I^m|X_R^m,Y^m) + H(iX_I^m|X_R^m,g(X_R^m) + iZ_I^m,Y^m)\big] \\ &\geq M\big[H(Y^m) + H(X_R^m|Y^m) + H(iX_I^m|X_R^m,iZ_I^m,Y^m)\big]. \end{aligned} \tag{35}$$

As in Lemma A.2, this lower bound is achievable when there exists a deterministic function, $\phi(\cdot)$, relating Z_R^m and Z_I^m, wherein we have $H(iZ_I^m|X_R^m,Y^m) = H(i\phi(f(X_R^m))|X_R^m) = 0$. We formalize this in Lemma A.3, where $\phi(\cdot)$ is bijective.

Lemma A.3 *Consider $X^m = X_R^m + iX_I^m \in \mathbb{C}^{dN}$, with $Y^m \in \mathbb{C}^k$ and $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$. We have that $H(S) \geq M[H(Y^m) + H(X_R^m|Y^m) + H(iX_I^m|X_R^m,iZ_I^m,Y^m)]$, with equality if the following condition holds $\forall m \in \mathcal{V}$:*

$$\begin{aligned} &\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN} : Z_I^m = \phi(Z_R^m) \implies \\ &H(S) = M\big[H(Y^m) + H(X_R^m|Y^m) + H(iX_I^m|X_R^m,iZ_I^m,Y^m)\big]. \end{aligned} \tag{36}$$

Recalling the expansion of $I(S;\theta_{\psi_S}) = H(S) - H(S|\theta_{\psi_S})$, we now focus on the $-H(S|\theta_{\psi_S})$ term:

$$\begin{aligned} -H(S|\theta_{\phi^S}) &= -H\Big((X^0,Y^0),(X^1,Y^1),\ldots,(X^{M-1},Y^{M-1})\Big|\theta_{\psi_S}\Big) \\ &= -\sum_{m=0}^{M-1} H\Big(X^m,Y^m\Big|(X^{m-1},Y^{m-1}),\ldots,(X^0,Y^0),\theta_{\psi_S}\Big) \\ &= -\sum_{m=0}^{M-1} H(X^m,Y^m|\theta_{\psi_S}) = -M\big[H(X^m,Y^m|\theta_{\psi_S})\big]. \end{aligned} \tag{37}$$

Paralleling Eq. (34), we note that Eq. (37) also follows from Section 2, since $S \sim P^{\otimes M}$. Accordingly, $H(X^i,Y^i|X^j,Y^j,\theta_{\psi_S}) = H(X^i,Y^i|\theta_{\psi_S})$, $\forall i,j \in \mathcal{V}$. Expanding $H(X^m,Y^m|\theta_{\psi_S})$, we have that:

$$\begin{aligned} -H(S|\theta_{\psi_S}) &= -M\big[H(X^m|\theta_{\psi_S}) + H(Y^m|X^m,\theta_{\psi_S})\big] \\ &= -M\big[H(X^m|\theta_{\psi_S}) + H(Y^m|X_R^m + iX_I^m,\theta_{f_S},\theta_{g_S})\big] \\ &= -M\big[H(X^m|\theta_{\psi_S}) - H(X_R^m + iX_I^m,\theta_{f_S},\theta_{g_S}) + H(Y^m,X_R^m + iX_I^m,\theta_{f_S},\theta_{g_S})\big] \\ &= -M\big[H(X^m|\theta_{\psi_S}) - H(X_R^m + iX_I^m,\theta_{g_S}) - H(\theta_{f_S}|X_R^m + iX_I^m,\theta_{g_S}) \\ &\quad + H(Y^m,X_R^m + iX_I^m,\theta_{g_S}) + H(\theta_{f_S}|Y^m,X_R^m + iX_I^m,\theta_{g_S})\big]. \end{aligned} \tag{38}$$

We now consider the $H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, \theta_{g_S}) - H(\theta_{f_S}|X_R^m + iX_I^m, \theta_{g_S})$ term. By the properties of conditional entropy, we observe that:

$$H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, \theta_{g_S}) - H(\theta_{f_S}|X_R^m + iX_I^m, \theta_{g_S}) \leq 0. \tag{39}$$

Where equality follows if $H(\theta_{f_S}|X_R^m + iX_I^m, \theta_{g_S}) = H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, \theta_{g_S})$. We now show that this condition is met when θ_{f_S} is deterministic given θ_{g_S} and $X_R^m + iX_I^m$.

Suppose we are given $X_R^m + iX_I^m$ and θ_{g_S}. It follows that $Z_R^m = g_S(X_R^m)$ is deterministic. We now impose the constraint presented in Lemma A.2 and A.3, wherein there exists a bijective, deterministic function, $\phi(\cdot)$, such that $Z_I^m = \phi(Z_R^m)$, $\forall m \in \mathcal{V}$. It follows that $Z_I^m = \phi(g_S(X_R^m))$ is deterministic. Per Eq. (39), the following equalities now hold under the imposed constraint:

$$Z_I^m = \phi(Z_R^m) \implies H(\theta_{f_S}|X_R^m + iX_I^m, \theta_{g_S}) = H(\theta_{f_S}|X_R^m + iX_I^m, Z_I^m), \tag{40}$$

$$Z_I^m = \phi(Z_R^m) \implies H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, \theta_{g_S}) = H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, Z_I^m). \tag{41}$$

We also recall the Markov chain presented in Figure 2. The Steinmetz network architecture informs us that Y^m does not reduce the uncertainty in θ_{f_S} given X_I^m and Z_I^m. Therefore, we have that:

$$Z_I^m = \phi(Z_R^m) \implies H(\theta_{f_S}|Y^m, X_R^m + iX_I^m, \theta_{g_S}) - H(\theta_{f_S}|X_R^m + iX_I^m, \theta_{g_S}) = 0, \tag{42}$$

$$\begin{aligned} -H(S|\theta_{\psi_S}) &\geq -M\big[H(X^m|\theta_{\psi_S}) - H(X_R^m + iX_I^m, \theta_{g_S}) + H(Y^m, X_R^m + iX_I^m, \theta_{g_S})\big] \\ &= -M\big[H(X^m|\theta_{\psi_S}) - H(Y^m|X_R^m + iX_I^m, \theta_{g_S})\big]. \end{aligned} \tag{43}$$

We summarize the achievability of this lower bound in Lemma A.4.

Lemma A.4 *Consider $X^m = X_R^m + iX_I^m \in \mathbb{C}^{dN}$, $Y^m \in \mathbb{C}^k$ and $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$, with $\theta_{\psi_S} \in \mathbb{R}^c, \theta_{g_S} \in \mathbb{R}^{c_2}$. We have that $-H(S|\theta_{\psi_S}) \geq -M[H(X^m|\theta_{\psi_S}) - H(Y^m|X_R^m + iX_I^m, \theta_{g_S})]$, with equality if the following condition holds $\forall m \in \mathcal{V}$:*

$$\begin{aligned} &\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN} : Z_I^m = \phi(Z_R^m) \implies \\ &-H(S|\theta_{\psi_S}) = -M\big[H(X^m|\theta_{\psi_S}) - H(Y^m|X_R^m + iX_I^m, \theta_{g_S})\big]. \end{aligned} \tag{44}$$

We can now determine the overall lower bound, $\mathcal{D}(Z^m)$, on $[I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S})]$ by substituting Eq. (32), (35), and (43) into Eq. (13):

$$\mathcal{D}(Z^m) \leq I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S}), \tag{45}$$

$$\begin{aligned} \text{where: } \mathcal{D}(Z^m) = H(Z_R^m) - I(Y^m; Z^m) - M\big[&H(X^m|\theta_{\psi_S}) - H(Y^m|X_R^m + iX_I^m, \theta_{g_S}) \\ &- H(Y^m) - H(X_R^m|Y^m) - H(iX_I^m|X_R^m, iZ_I^m, Y^m)\big]. \end{aligned}$$

Revisiting Corollary 4.2, we note that $\mathcal{D}(Z^m)$ is an achievable lower bound, with equality observed when the imposed condition, $f \in \mathcal{F}^m$, delineates Z_I^m and Z_R^m as being related by a deterministic, bijective function, $\phi(\cdot)$. We summarize this result in Lemma A.5.

Lemma A.5 *Consider $X^m = X_R^m + iX_I^m \in \mathbb{C}^{dN}$, $Y^m \in \mathbb{C}^k$, and $Z^m = Z_R^m + iZ_I^m \in \mathbb{C}^{lN}$ with $\theta_{\psi_S} \in \mathbb{R}^c, \theta_{g_S} \in \mathbb{R}^{c_2}$. It follows that $\mathcal{D}(Z^m) \leq I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S})$, where:*

$$\begin{aligned} \mathcal{D}(Z^m) = H(Z_R^m) - I(Y^m; Z^m) - M\big[&H(X^m|\theta_{\psi_S}) - H(Y^m|X_R^m + iX_I^m, \theta_{g_S}) \\ &- H(Y^m) - H(X_R^m|Y^m) - H(iX_I^m|X_R^m, iZ_I^m, Y^m)\big]. \end{aligned} \tag{46}$$

With equality if the following condition holds $\forall m \in \mathcal{V}$:

$$\begin{aligned} &\forall Z_I^m \in \mathbb{R}^{lN},\ \exists! Z_R^m \in \mathbb{R}^{lN} : Z_I^m = \phi(Z_R^m) \implies \\ &I(X^m; Z^m) - I(Y^m; Z^m) + I(S; \theta_{\psi_S}) = \mathcal{D}(Z^m). \end{aligned} \tag{47}$$

This result is also summarized in Theorem 4.3 of the main text. We extend this result to derive the smaller upper bound on the generalization gap, Δs, provided in Theorem 4.4 of the main text.

B Additional Proofs

B.1 Complementarity Principle

For completeness, we prove Corollary 3.1 from the main text. Per the notation outlined in Section 3.1, we first note the expansions of the auto-covariance matrices, $\mathbf{K}_{X_R^m}$ and $\mathbf{K}_{X_I^m}$:

$$\mathbf{K}_{X_R^m} = \begin{bmatrix} \mathbf{K}_{Z_R^m} & \mathbf{K}_{Z_R^m,\Lambda_R^m} \\ \mathbf{K}_{\Lambda_R^m,Z_R^m} & \mathbf{K}_{\Lambda_R^m} \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{Z_R^m} & \mathbf{0}_{kN\times(d-k)N} \\ \mathbf{0}_{(d-k)N\times kN} & \mathbf{K}_{\Lambda_R^m} \end{bmatrix}, \tag{48}$$

$$\mathbf{K}_{X_I^m} = \begin{bmatrix} \mathbf{K}_{Z_I^m} & \mathbf{K}_{Z_I^m,\Lambda_I^m} \\ \mathbf{K}_{\Lambda_I^m,Z_I^m} & \mathbf{K}_{\Lambda_I^m} \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{Z_I^m} & \mathbf{0}_{kN\times(d-k)N} \\ \mathbf{0}_{(d-k)N\times kN} & \mathbf{K}_{\Lambda_I^m} \end{bmatrix}. \tag{49}$$

Regarding the cross-covariance matrices, $\mathbf{K}_{X_I^m,X_R^m} = \mathbf{K}_{X_R^m,X_I^m}^T$ and $\overline{\mathbf{K}}_{X_I^m,X_R^m} = \overline{\mathbf{K}}_{X_R^m,X_I^m}^T$, where:

$$\mathbf{K}_{X_R^m,X_I^m} = \begin{bmatrix} \mathbf{K}_{Z_R^m,Z_I^m} & \mathbf{K}_{Z_R^m,\Lambda_I^m} \\ \mathbf{K}_{\Lambda_R^m,Z_I^m} & \mathbf{K}_{\Lambda_R^m,\Lambda_I^m} \end{bmatrix}, \quad \overline{\mathbf{K}}_{X_R^m,X_I^m} = \begin{bmatrix} \mathbf{0}_{kN\times kN} & \mathbf{K}_{Z_R^m,\Lambda_I^m} \\ \mathbf{K}_{\Lambda_R^m,Z_I^m} & \mathbf{K}_{\Lambda_R^m,\Lambda_I^m} \end{bmatrix}. \tag{50}$$

We now derive and compare the $L_{p,q}$ norm of $\mathbf{\Sigma_J}$ and $\mathbf{\Sigma_S}$. We first note that $\mathbf{K}_{Z_R^m,Z_I^m} = \mathbf{0}_{kN\times kN}$ for the separate-then-joint processing case, since $f(\cdot)$ and $g(\cdot)$ do not consider the interactions between Z_R^m and Z_I^m. These interactions are not lost, however, as they are leveraged by $h^*(\cdot)$ during the joint processing step. Accordingly, it follows that:

$$\begin{aligned}
\|\mathbf{\Sigma_J}\|_{p,q} &= \left(\sum_{i=1}^{dN} \left(\|[\mathbf{K}_{X_R^m}]_i\|_q^q + \|[\mathbf{K}_{X_R^m,X_I^m}]_i\|_q^q \right)^{\frac{p}{q}} + \sum_{i=1}^{dN} \left(\|[\mathbf{K}_{X_I^m,X_R^m}]_i\|_q^q + \|[\mathbf{K}_{X_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \right)^{\frac{1}{p}} \\
&= \Bigg(\sum_{i=1}^{kN} \left(\|[\mathbf{K}_{Z_R^m}]_i\|_q^q + \|[\mathbf{K}_{Z_R^m,Z_I^m}]_i\|_q^q + \|[\mathbf{K}_{Z_R^m,\Lambda_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{(d-k)N} \left(\|[\mathbf{K}_{\Lambda_R^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_R^m,Z_I^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_R^m,\Lambda_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{kN} \left(\|[\mathbf{K}_{Z_I^m}]_i\|_q^q + \|[\mathbf{K}_{Z_I^m,Z_R^m}]_i\|_q^q + \|[\mathbf{K}_{Z_I^m,\Lambda_R^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{(d-k)N} \left(\|[\mathbf{K}_{\Lambda_I^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_I^m,Z_R^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_I^m,\Lambda_R^m}]_i\|_q^q \right)^{\frac{p}{q}} \Bigg)^{\frac{1}{p}} \\
&\geq \Bigg(\sum_{i=1}^{kN} \left(\|[\mathbf{K}_{Z_R^m}]_i\|_q^q + \|[\mathbf{K}_{Z_R^m,\Lambda_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{(d-k)N} \left(\|[\mathbf{K}_{\Lambda_R^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_R^m,Z_I^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_R^m,\Lambda_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{kN} \left(\|[\mathbf{K}_{Z_I^m}]_i\|_q^q + \|[\mathbf{K}_{Z_I^m,\Lambda_R^m}]_i\|_q^q \right)^{\frac{p}{q}} \\
&\quad + \sum_{i=1}^{(d-k)N} \left(\|[\mathbf{K}_{\Lambda_I^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_I^m,Z_R^m}]_i\|_q^q + \|[\mathbf{K}_{\Lambda_I^m,\Lambda_R^m}]_i\|_q^q \right)^{\frac{p}{q}} \Bigg)^{\frac{1}{p}} \\
&= \left(\sum_{i=1}^{dN} \left(\|[\mathbf{K}_{X_R^m}]_i\|_q^q + \|[\overline{\mathbf{K}}_{X_R^m,X_I^m}]_i\|_q^q \right)^{\frac{p}{q}} + \sum_{i=1}^{dN} \left(\|[\overline{\mathbf{K}}_{X_I^m,X_R^m}]_i\|_q^q + \|[\mathbf{K}_{X_I^m}]_i\|_q^q \right)^{\frac{p}{q}} \right)^{\frac{1}{p}} \\
&= \|\mathbf{\Sigma_S}\|_{p,q}
\end{aligned}$$

Therefore, $\|\mathbf{\Sigma_J}\|_{p,q} \geq \|\mathbf{\Sigma_S}\|_{p,q}$. We also note $\mathbf{K}_{Z_R^m,Z_I^m} = \mathbf{K}_{Z_I^m,Z_R^m} = \mathbf{0}_{dN\times dN}$ when $Z_R^m \perp\!\!\!\perp Z_I^m$. Accordingly, it follows that $Z_R^m \perp\!\!\!\perp Z_I^m \implies \|\mathbf{\Sigma_J}\|_{p,q} = \|\mathbf{\Sigma_S}\|_{p,q}$.

B.2 Orthogonality of Latent Analytic Signal Representation

We now prove Corollary 5.1 from the main text, which claims $Z_I^m = \mathcal{H}\{Z_R^m\}$ enforces orthogonality between Z_R^m and Z_I^m. Let $\mathbf{F}_R^m = \mathcal{F}\{Z_R^m\} \in \mathbb{C}^{lN}$ denote the DFT of Z_R^m and let $\mathbf{H}_R^m \in \mathbb{C}^{lN}$ denote the frequency components of $\mathcal{H}\{Z_R^m\}$. We consider the inner product $\langle \cdot, \cdot \rangle : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}_{\geq 0}$, wherein:

$$\begin{aligned}\langle Z_R^m, Z_I^m \rangle &= \mathbb{E}[Z_R^m Z_I^m] = \mathbb{E}[Z_R^m \ \mathcal{H}\{Z_R^m\}] = \mathbb{E}\left[\sum_{n=0}^{lN-1} Z_R^m[n]\mathcal{H}\{Z_R^m\}[n]\right] \\ &= \mathbb{E}\left[\sum_{n=0}^{lN-1} Z_R^m[n]\left(\frac{1}{lN}\sum_{b=0}^{lN-1} \mathbf{H}_R^m[b] e^{\frac{i2\pi bn}{lN}}\right)\right] \end{aligned} \tag{51}$$

We now substitute Eq. (20) into the above expression, and expand the $\mathbf{H}_R^m[b]$ term.

$$\begin{aligned}\langle Z_R^m, Z_I^m \rangle &= \mathbb{E}\left[\sum_{n=0}^{lN-1} Z_R^m[n]\left(\frac{1}{lN}\sum_{b=0}^{lN-1} \mathbf{F}_R^m[b] \cdot (\pm i) e^{\frac{i2\pi bn}{lN}}\right)\right] \\ &= \mathbb{E}\left[\frac{\pm i}{lN}\sum_{b=0}^{lN-1} \mathbf{F}_R^m[b] \sum_{n=0}^{lN-1} Z_R^m[n] e^{\frac{i2\pi bn}{lN}}\right] \\ &= \mathbb{E}\left[\frac{\pm i}{lN}\sum_{b=0}^{lN-1} \mathbf{F}_R^m[b]\mathbf{F}_R^m[b]^*\right] = \mathbb{E}\left[\frac{\pm i}{lN}\sum_{b=0}^{lN-1} |\mathbf{F}_R^m[b]|^2 \text{sgn}(b)\right] \\ &= 0. \end{aligned} \tag{52}$$

Since $\text{sgn}(b)$ is odd and $|\mathbf{F}_R^m[b]|^2$ is even, the sum evaluates to zero. Therefore, when $Z_I^m = \mathcal{H}\{Z_R^m\}$, $\langle Z_R^m, Z_I^m \rangle = 0$, and consequently, Z_R^m and Z_I^m are orthogonal.

C Dataset Descriptions

C.1 CV-MNIST Dataset

The **MNIST** dataset is a collection of handwritten digits commonly used to train image processing systems. The CV-MNIST dataset is formed by taking a $dN = 784$-point DFT of each flattened MNIST image, yielding a 728-dimensional real feature vector, and a 728-dimensional imaginary feature vector. For the classification result in Section 6.1, we consider the first $M = 500$ training samples from CV-MNIST and for the noise robustness result, we consider $M = 60{,}000$ training samples from CV-MNIST. We consider 10,000 test samples in both cases. The features and labels within CV-MNIST are summarized as follows:

- Each feature (image) is represented as a 728-dimensional real vector and a 728-dimensional imaginary vector, obtained by taking the DFT of the original size 28×28 grayscale image.
- Target Variable: The numerical class (digit) the image represents, ranging from 1 to 10 ($k = 10$).

C.2 CV-CIFAR-10 Dataset

The **CIFAR-10** dataset is a collection of color images categorized into 10 different classes, and is commonly used to train image processing systems. The CV-CIFAR-10 dataset is formed by taking a $dN = 3072$-point DFT of each flattened CIFAR-10 image, yielding a 3072-dimensional real feature vector, and a 3072-dimensional imaginary feature vector. For the classification and noise robustness results from Section 6.1, we consider $M = 50{,}000$ training samples and 10,000 test samples from CV-CIFAR-10. The features and labels within CV-CIFAR-10 are summarized as follows:

- Each feature (image) is represented as a 3072-dimensional real vector and a 3072-dimensional imaginary vector, obtained by taking the DFT of the original size $32 \times 32 \times 3$ image.
- Target Variable: The numerical class (category) the image represents, ranging from 1 to 10 ($k = 10$).

C.3 CV-CIFAR-100 Dataset

The **CIFAR-100** dataset is a collection of color images categorized into 100 different classes, and is commonly used to train image processing systems. The CV-CIFAR-100 dataset is formed by taking a $dN = 3072$-point DFT of each flattened CIFAR-100 image, yielding a 3072-dimensional real feature vector and a 3072-dimensional imaginary feature vector. For the classification result from Section 6.1, we consider $M = 50{,}000$ training samples and 10,000 test samples from CV-CIFAR-100. The features and labels within CV-CIFAR-100 are summarized as follows:

- Each feature (image) is represented as a 3072-dimensional real vector and a 3072-dimensional imaginary vector, obtained by taking the DFT of the original size $32 \times 32 \times 3$ image.
- Target Variable: The numerical class (category) the image represents, ranging from 1 to 100 ($k = 100$).

C.4 CV-FSDD Dataset

The **FSDD** (Free Spoken Digit Dataset) is a collection of audio recordings of spoken digits categorized into 10 different classes, and is commonly used to train audio processing systems. The CV-FSDD dataset is formed by taking a $dN = 8000$-point DFT of each flattened FSDD recording, yielding a 8000-dimensional real feature vector and a 8000-dimensional imaginary feature vector. For the classification result from Section 6.1, we consider $M = 2{,}700$ training samples and 300 test samples from CV-FSDD. The features and labels within CV-FSDD are summarized as follows:

- Each feature (audio recording) is represented as a 8000-dimensional real vector and a 8000-dimensional imaginary vector, obtained by taking the DFT of the original audio signal.
- Target Variable: The numerical digit (category) the recording represents, ranging from 1 to 10 ($k = 10$).

C.5 RASPNet Dataset

The **RASPNet** dataset for radar adaptive signal processing consists of radar returns gathered from 100 realistic radar scenarios located in the contiguous United States. We consider the scenario index $i = 29$ from RASPNet, which corresponds to the Bonneville Salt Flats, UT, and consider the real and imaginary parts of a size $5 \times 21 \times 16$ radar return, comprising 5 realizations, 21 range bins, and 16 channels (single pulse transmission). Accordingly, each sample is defined by a $dN = 1680$-dimensional real feature vector, and a 1680-dimensional imaginary feature vector. Each radar return contains a single point target, with position encoded in Cartesian coordinates (x, y). For the regression result from Section 6.1, we consider $M = 20{,}000$ training samples and 5,000 test samples from scenario $i = 29$, which can be accessed here. The features and labels are summarized as follows:

- Each feature (radar return) posits a 1680-dimensional real vector and a 1680-dimensional imaginary vector.
- Target Variable: The target location in Cartesian coordinates (x, y) ($k = 2$).

D Neural Network Architectures

We provide a detailed description of three different neural network architectures designed for classification and regression. Each of these architectures were employed to generate the respective empirical results pertaining to the aforementioned tasks.

D.1 Steinmetz Neural Network

The Steinmetz Network is designed to handle both real and imaginary components of the input data separately before combining them for the final prediction. This architecture can be applied to both classification and regression tasks (see Figure 5).

- **Fully Connected Layer (`realfc1`)**: Transforms the real part of the input features to a higher dimensional space. It takes dN-dimensional inputs and yields lN-dimensional outputs.

- **ReLU Activation** (`realrelu1`): Introduces non-linearity to the model. It operates element-wise on the output of `realfc1`.
- **Fully Connected Layer** (`realfc2`): Further processes the output of `realrelu1`, yielding lN-dimensional outputs.
- **ReLU Activation** (`realrelu2`): Applies non-linearity to the output of `realfc2`.
- **Fully Connected Layer** (`imagfc1`): Transforms the imaginary part of the input features to a higher dimensional space, paralleling `realfc1`.
- **ReLU Activation** (`imagrelu1`): Applies non-linearity to the output of `imagfc1`.
- **Fully Connected Layer** (`imagfc2`): Further processes the output of `imagrelu1`, yielding lN-dimensional outputs.
- **ReLU Activation** (`imagrelu2`): Applies non-linearity to the output of `imagfc2`.
- **Fully Connected Layer** (`regressor`): Combines the extracted features from both networks into a single $2lN$-dimensional feature vector (the latent space), which is then passed through a fully connected layer to produce the final output of dimension k.

D.2 Real-Valued Neural Network

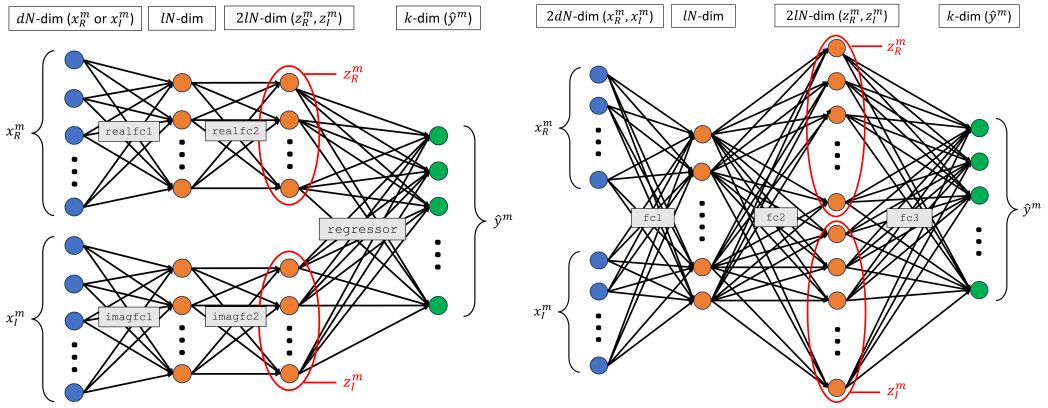
The Real-valued neural network (RVNN) architecture is a straightforward and effective approach for handling both real and imaginary components by concatenating them and processing them together. It can be used for both classification and regression tasks (see Figure 5).

- **Fully Connected Layer** (`fc1`): Takes the concatenated real and imaginary components ($2dN$-dimensional) as input and produces lN-dimensional features.
- **ReLU Activation** (`relu1`): Introduces non-linearity to the model after `fc1`.
- **Fully Connected Layer** (`fc2`): Further processes the output of `relu1`, yielding the $2lN$-dimensional latent space as the output.
- **ReLU Activation** (`relu2`): Applies non-linearity to the output of `fc2`.
- **Fully Connected Layer** (`fc3`): Produces the final output of dimension k.

D.2.1 Complex-Valued Neural Network

The Complex-Valued Neural Network (CVNN) is designed to handle complex-valued data by treating the real and imaginary parts jointly as complex numbers. It can be used in classification and regression tasks (see Figure 6). For classification tasks, we take the magnitude of the `fc3` layer output.

- **Complex Linear Layer** (`fc1`): Transforms the dN-dimensional complex inputs into lN-dimensional complex features.
- **Complex ReLU Activation** (`relu1`): Applies complex-valued ReLU activation after `fc1`.
- **Complex Linear Layer** (`fc2`): Further processes the lN-dimensional complex features into lN-dimensional complex features (the latent space).
- **Complex ReLU Activation** (`relu2`): Applies complex-valued ReLU activation after `fc2`.
- **Complex Linear Layer** (`fc3`): Produces the final k-dimensional output by transforming the lN-dimensional complex features.



(a) Steinmetz neural network architecture **(b)** Real-valued neural network architecture

Figure 5: Real-valued architectures for complex-valued data processing.

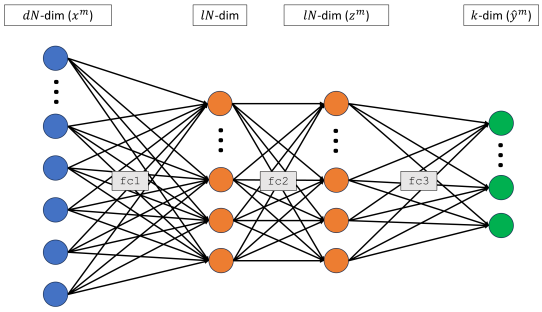


Figure 6: Complex-valued neural network architecture.

D.3 Neural Network Training Hyperparameters

The relevant hyperparameters used to train the neural networks from Appendix Section D are provided in Table 3. All results presented in the main text were produced using these hyperparameter choices.

Table 3: Neural network training hyperparameters (grouped by dataset/task).

Dataset/Task	**Experiment**	**Optimizer**	**Learning Rate (α)**	**Consistency penalty (β)** (analytic neural network)
CV-MNIST	No ablations	Adam	0.001	0.001
CV-MNIST	Additive noise	Adam	0.001	0.001
CV-CIFAR-10	No ablations	Adam	0.0001	0.001
CV-CIFAR-10	Additive noise	Adam	0.0001	0.001
CV-CIFAR-100	No ablations	Adam	0.0001	0.001
CV-FSDD	No ablations	Adam	0.0002	1.0e-6
RASPNet	No ablations	Adam	0.005	0.001
Channel Identification	SNR = 5 dB	Adam	0.0001	0.0001