Training Deep Neural Networks with HSIC and Backpropagation

Roshan Birjais Kevin I-Kai Wang Waleed Abdulla
The University of Auckland, New Zealand
E-mail: rsof418@aucklanduni.ac.nz E-mail: kevin.wang@auckland.ac.nz E-mail: w.abdulla@auckland.ac.nz

***Abstract*—Deep Learning has made significant strides in recent years, particularly in supervised learning tasks, leading to the development of numerous architectures aimed at improving various aspects of model performance. Despite the effectiveness of backpropagation (BP) and stochastic gradient descent in training deep networks, these methods are often hindered by time-intensive computations, exploding and vanishing gradients, and significant memory overhead. Alternative training strategies that reduce reliance on global BP are increasingly being explored to address these limitations. This paper proposes a simple architecture that integrates Hilbert Schmidt Independence Criterion (HSIC) layers with linear layers, where the HSIC layers are trained locally, and the linear layers are optimized using global BP. This hybrid approach mitigates the drawbacks of BP while enhancing the model's ability to learn complex features across multiple layers. Our proposed model is benchmarked against existing HSIC-only models across several datasets, including MNIST, CIFAR-10, and Fashion MNIST. Results demonstrate the superior performance of our model, in terms of accuracy achieved and memory used. Additionally, we demonstrate its robustness and effectiveness when handling noisy data. The code is available at https://github.com/AreeBeee/HSIC.git**

I. Introduction

In recent years, Deep Learning has emerged as a prominent area of research due to its remarkable success as a machine learning technique, particularly in supervised learning tasks. Numerous upgraded versions of architectures have been developed, each striving to improve upon its predecessors in various aspects. Discovering efficient methods to train a model has been a central concern for every researcher in this domain. Although training a model using backpropagation and stochastic gradient descent, or its variants, is effective, it proves to be highly time-consuming, particularly when applied to deep networks. Additionally, challenges such as exploding gradients, vanishing gradients, and the need for update locking further compound the issue. Figure 1 illustrates how training commences with BP which has two passes. In forward propagation, input data is processed sequentially through successive network layers to generate output. This output is subsequently employed to evaluate the loss function, which quantifies the divergence between the predicted outputs and the actual target values. In the subsequent backward propagation phase, the loss function's gradients with respect to each parameter are calculated using the chain rule, a process streamlined by automatic differentiation in modern computational frameworks. BP is the most conventional method used in supervised learning. However, the limitations of BP such as vanishing gradients, getting stuck in local minima especially when the network is deep, and also the memory overhead are the key problems.

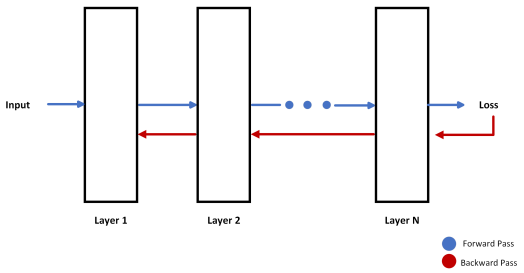

Fig. 1. Illustration of the forward pass and backward pass in a neural network. This process enables the network to learn from the training data and improve its performance over time.

A need to find an alternative that eliminates the necessity for global BP while maintaining efficiency is required. This involves focusing on information approximation, where only relevant information passes through the layers initially. A significant advancement in this direction was demonstrated by Naftali and Noga, highlighting the information-theoretic tradeoff between compression and relevance and demonstrating that the quantification of Deep Neural Networks (DNNs) can be achieved through the mutual information between the layers and the inputs and outputs.[1] Mutual information[2] is used to find the dependence between two random variables x and y, and can be expressed using Shannon's entropy as $I(x, y) = H(y) - H(y|x)$ where $H(y|x)$ is the conditional entropy. Considering x as the representation of the input and y being the label in a network, the mutual information offers a training objective that engages with x directly without involving BP through cross-entropy. Calculating mutual information is difficult [3], Ma et al [4] substituted mutual information with the Hilbert Schmidt Information Criterion (HSIC) for training the model without using backpropagation and cross-entropy loss. The HSIC layers help to mitigate the limitations of BP and maintain the performance of the network. It addresses the issues of exploding and vanishing gradients, thereby enabling the training of very deep networks without the need for skip connections. Additionally, it eliminates the necessity for symmetric feedback or update locking. However, using HSIC increases the memory overhead. Calculating HSIC involves computing kernel matrices with a complexity of $O(n^2)$, which can be intensive for large datasets. In contrast, backpropagation

has a complexity of $O(nm)$, where n is the number of samples and m is the number of parameters, making it more scalable for larger datasets. Also, the indirect nature of HSIC in promoting dependence can lead to slower convergence rates compared to direct optimization of the loss function using backpropagation. We introduce a model where after every HSIC trained layer, a fully connected layer is introduced. These alternate linear layers are trained using global BP. Training done in this way allows a deep network to integrate features learned by lower layers into more complex ones in higher layers. This is evident from the enhanced accuracy of the model.

Deep neural networks trained on large, well-annotated datasets have achieved remarkable results in image classification and other tasks. However, the collection of such high-quality datasets can be both time-consuming and costly, leading to a growing interest in leveraging larger but noisier datasets that are more readily available. In this paper, we have evaluated our model for robustness and noise adaptation, showing how well the model maintains its performance under noisy conditions, demonstrating its resilience and reliability.. In Section 2, we provide background information, detailing the techniques employed to design the architecture. Section 3 delves into the architecture's structure and operation, offering a comprehensive overview of its components and functionalities. Finally, in Section 4, we present the results obtained, providing a detailed analysis of its performance and effectiveness. The contributions of this research are as follows:

- This research introduces a novel approach that combines the Hilbert Schmidt Independence Criterion (HSIC) without backpropagation (BP) and traditional BP for training deep neural networks (DNNs). The proposed architecture demonstrates similar performance to existing models while significantly reducing memory overhead.
- The network trained using this approach exhibits improved resilience to noise. The model has been tested for robustness and noise adaptation, highlighting its effectiveness in maintaining performance under noisy conditions.

II. Background

In this section, we present the foundational background, outlining the various techniques utilized in the design of the architecture.

A. Relative Entropy and Mutual Information

The entropy of a random variable is the uncertainty associated with it, representing the average amount of information needed to describe it, and the measure of distance between two distributions $p(x)$ and $p(y)$ is the relative entropy. Mutual information represents a specific instance within it. Mutual information serves as a comprehensive metric for gauging the relationship between two random variables. In a discrete domain, it quantifies the dependence between two random variables X and Y.

B. Hilbert Schmidt Independence Criterion

HSIC is a way to calculate mutual information between two distributions. Consider F, a Reproducing Kernel Hilbert Space (RKHS) of functions mapping from X to $\mathbb{R}$, where X is a separable metric space, and $\mathbb{R}$ represents the set of real numbers. For any two points x and x' in X, we can associate elements $\phi(x)$ and $\phi(x')$ in F (referred to as feature mapping) such that the associated reproducing F kernel $k(x, x') = \langle \phi(x), \phi(x') \rangle$.

In Reproducing Kernel Hilbert Space (RKHS), HSIC can be understood as a measure of the distance between the embeddings of the joint distribution P_{XY} and the product of the marginal distributions $P_X P_Y$, where X and Y are the random variables. When characteristic RKHSs are employed, the independence of the random variables can be inferred by checking if the value of HSIC is zero. Compared to other traditional independence measures, HSIC offers multiple advantages, including its ease of computation, quick convergence, and minimal estimation bias when working with limited data samples. HSIC-based learning methods revolve around the concept of leveraging HSIC to gauge the correlation or linkage between the various elements being analyzed, like the inputs and outputs. The objective is to optimize the solutions by either increasing or decreasing these associations. Let X and Y be the random variables, $k(\cdot, \cdot)$ and $l(\cdot, \cdot)$ the kernel functions, then

$$\begin{aligned} \mathrm{HSIC}(X, Y) = \; & \mathbb{E}_{(x,x',y,y')}[k(x, x')l(y, y')] \\ & + \mathbb{E}_{(x,x')}[k(x, x')] \cdot \mathbb{E}_{(y,y')}[l(y, y')] \\ & - 2\mathbb{E}_{(x,y)} \left[\mathbb{E}_{(x')}[k(x, x')] \cdot \mathbb{E}_{y}[l(y, y')] \right] \end{aligned} \tag{1}$$

Where $\mathbb{E}_{(x,x',y,y')}$ denote expectation over independent pairs (x, y) and x', y'. A kernel function transforms the input data into a feature space with a high dimension, where the statistical correlation between two random variables can be more conveniently assessed.

III. Methodology

In this section, we introduce the proposed architecture. As shown in Figure 2, the proposed architecture consists of one or many sub-networks. Generally, the sub-network (SN) consists of one or more HSIC layers followed by one or more linear layers. The number of SNs used depends on the problem, just like in any other neural network; the complexity of the task and data, modelling goals like generalisation and overfitting decide on what and how many layers are needed.

A. Datasets

This section contains information about the datasets used for the experimentation. 1) The MNIST dataset is a widely used benchmark in the field of Deep learning, consisting of 70,000 grayscale images of handwritten digits from 0 to 9, each sized at 28x28 pixels. It is split into 60,000 training images and 10,000 testing images. 2) The MNIST Fashion dataset serves as a more challenging alternative to MNIST, containing 70,000

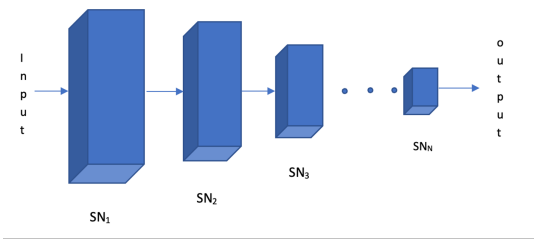

Fig. 2. Block diagram of the network which consists of one or more sub-networks depending the requirements of the data.

grayscale images of 10 different clothing items, also divided into a training set of 60,000 images and a test set of 10,000 images. 3) CIFAR-10 is another popular dataset comprising 60,000 32x32 color images in 10 classes, with 50,000 images for training and 10,000 for testing.

B. Proposed Architecture

In this research, we proposed a simple deep-learning architecture that combines HSIC layers with linear layers trained via BP. In Figure 3, represents the architecture, which is made up of sub-networks. We use linear layers with ReLu as the activation functions and softmax for the last layer. Each sub-network consists of one HSIC layer and one linear layer. It is better understood from Algorithm 1. We have tested the model on various datasets and have observed that the model performs better than the architecture trained using HSIC only found in the literature. Having a linear layer after every HSIC layer allows for capturing complex relationships in the data and transforming the features extracted by the HSIC layer into a higher-dimensional space, potentially enhancing the model's ability to learn and discriminate between different patterns in the data.

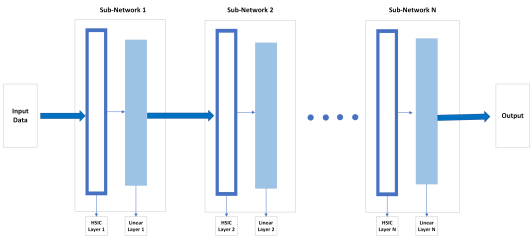

Fig. 3. Illustration the proposed network architecture in which the HSIC layers are trained without BP and linear layers are trained using BP.

Consider a neural network comprising L hidden layers, for the layers, denoted as $T_i(\cdot) : \mathbb{R}^{d_{i-1}} \rightarrow \mathbb{R}^{d_i}$, resulting in hidden representations $Z_i \in \mathbb{R}^{m \times d_i}$, where $i \in \{1, \ldots, L\}$, and m represents the batch size, then

$$Z_i = \arg\min_{Z_i} \left(nHSIC(Z_i, X) - \beta nHSIC(Z_i, Y)\right) \quad (2)$$

The input data $X \in \mathbb{R}^{m \times dx}$ and labels $Y \in \mathbb{R}^{m \times dy}$, where m denotes the batch size, i ranges from 0 to L, and L represents the total number of hidden layers. The dimensions dx and dy correspond to the input and output variables, respectively. The parameter β modulates the trade-off between the IB objectives that aims to compress input data while retaining as much relevant information as possible for predicting the output and nHSIC is given for each term is

$$nHSIC(Z_i, X) = \text{tr}(K_e Z_i K_e X) \quad (3)$$

$$nHSIC(Z_i, Y) = \text{tr}(K_e Z_i K_e Y) \quad (4)$$

SN_1 The above equations suggest that the optimal hidden representation Z_i achieves a delicate equilibrium between independence from irrelevant input details and association with the output. K_e is the centered kernel matrix derived from the original kernel matrix by removing the mean effects. The convergence of equation (3) ideally preserves the essential information required for label prediction while eliminating extraneous details that might lead to overfitting. The features derived from the HSIC layers serve as input to the BP-led layers. As the data traverses through the network comprising L layers, the final output is produced.

Algorithm 1 Training an MLP with HSIC and Backpropagation

Input: X (input data), Y (labels), n (number of samples), m (batch size), L (number of layers), α (learning rate), β (HSIC regularization parameter), T_{i-1} and F_i are parametrized by $\{\theta_i \mid W_i, b_i\}$ and $\{\phi_i \mid W_i, b_i\}$ respectively.
Initialize: θ_i (HSIC layer parameters), ϕ_i (BP layer parameters) for $i \in \{1, \ldots, L\}$
for $j \in \{1, \ldots, n/m\}$ **do**
  $Z_0 \leftarrow X_j$
  for $i \in \{1, \ldots, L\}$ **do**
    if $i \in$ HSIC Layers **then**
      $Z_i \leftarrow T_{i-1}(Z_{i-1})$
      $g_i \leftarrow \nabla_{\theta_i}(nHSIC(Z_i, X_j) - \beta \times nHSIC(Z_i, Y_j))$
      $\theta_i \leftarrow \theta_i - \alpha g_i$
    else
      $V_i \leftarrow F_i(Z_{i-1})$
      $h_i \leftarrow \nabla_{\phi_i}(\text{Loss}(V_i, Y_j))$
      $\phi_i \leftarrow \phi_i - \alpha h_i$
      $Z_i \leftarrow V_i$
    end if
  end for
end for

IV. Results

The performance of both models was assessed across several datasets, including MNIST, CIFAR-10, and Fashion MNIST. We evaluated and compared the results with those reported in [4] as shown in Figure 4. The observed accuracies from the proposed model highlight its effectiveness when compared to

models where layers are trained exclusively using HSIC. The testing results demonstrate that the proposed architecture outperforms the HSIC-only model, suggesting that incorporating linear layers between HSIC layers enhances the model's ability to learn and capture essential features for accurate prediction. This indicates that the integration of linear layers contributes to a more robust feature extraction process, ultimately leading to improved performance.

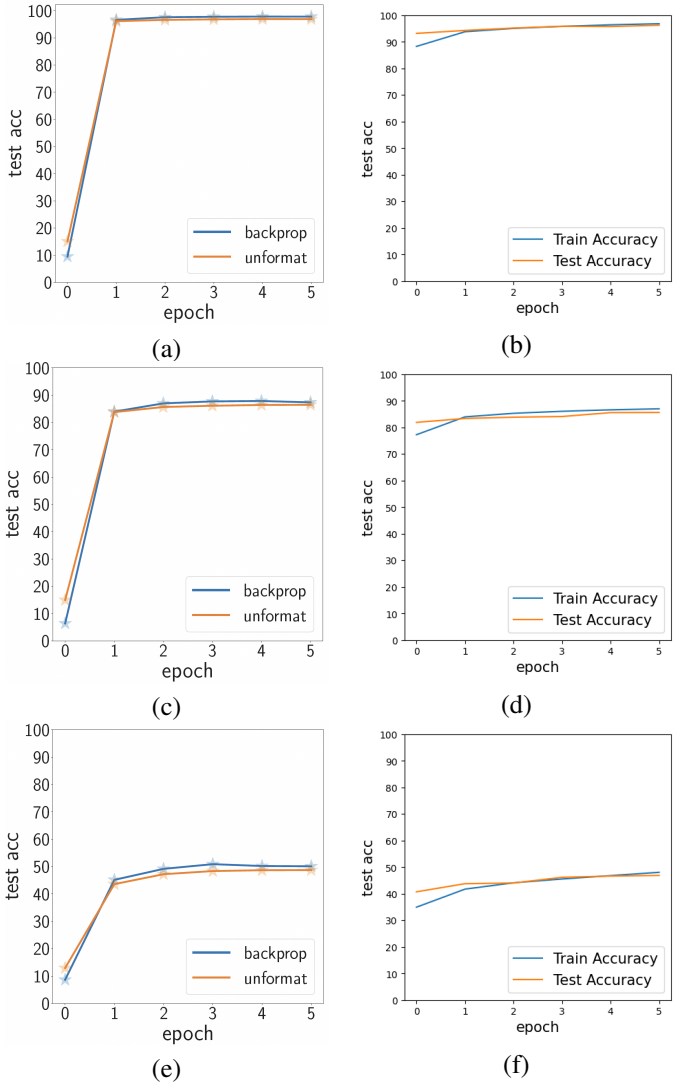

Fig. 4. Performance evaluation comparison between HSIC-trained architecture and our proposed method. The plots (a), (c), (e) are taken from [4]. Plots (b), (d), and (f) are the plots obtained from the proposed method. The first row shows the result for MNIST, the second row for FashionMNIST, and the third row for CIFAR10.

In addition to achieving high accuracy, computational efficiency is paramount for any model. We conducted a comparative analysis of architectures based on several key metrics number of layers, and nodes per layer, as detailed in Table I. The proposed model comprises two HSIC layers and two linear layers, structured as 714-512-256-128-10. This configuration is more compact compared to the 784-256-256-256-256-256-10 architecture described in [4]. Despite using only 3 hidden layers with fewer nodes per layer, our model achieves similar accuracy, indicating its efficiency in terms of memory usage.

TABLE I
COMPARATIVE ANALYSIS OF TWO ARCHITECTURES, DETAILING THE NUMBER OF LAYERS AND NODES USED IN EACH MODEL.

Method	Hidden Layers	Nodes	epochs
HSIC [4]	5	714-256-256-256-256-256-10	6
Proposed model	3	714-512-256-128-10	6

A. Experimental results with Noise

The presence of noise is a well-known issue that significantly impacts data quality, particularly in classification tasks [5], [6]. In various application domains such as medical image processing [7], speech recognition [8], and stock market analysis [9], classification accuracy is a crucial metric. Deep neural networks are usually trained through supervised learning with large, meticulously curated datasets that are free from noise.

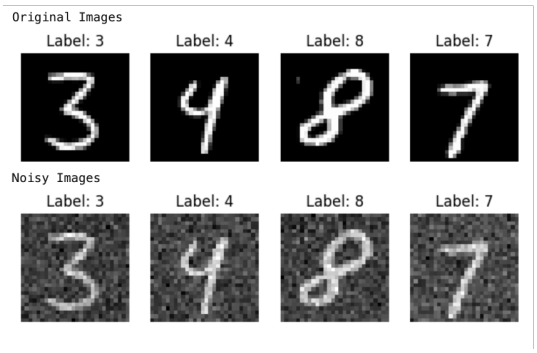

Fig. 5. Images before and after the Gaussian noise is added to it (standard deviation = 0.05).

However, the requirement for such datasets limits the range of problems that can be tackled because in real life, getting data that is clean is not most of the time true. This has resulted in a surge of deep learning studies focused on the same tasks using these familiar datasets. We utilized the MNIST dataset and introduced Gaussian noise to the data. Figure 5 shows the image before and after the noise is added to the images.

This approach allowed us to assess the performance of our model under conditions where data is not perfectly clean, providing insights into its robustness and generalisation. By simulating real-world scenarios where noise may be present, we aimed to better understand the model's effectiveness in practical applications.

The results indicate that our model exhibits greater robustness. This enhanced resilience suggests that the model is capable of maintaining high performance even when confronted with noisy or imperfect data, which is critical for its deployment in real-world environments where data quality can vary. Table II indicates that there is minimal difference in the model's accuracy when trained and tested on noisy data. This

TABLE II
PERFORMANCE COMPARISON OF PROPOSED METHOD WHEN BOTH TRAINING AND TESTING DATA IS NOISY.

	Std = 0.05	**Std = 0.5**	**Std = 0.8**
Train Accuracy	96.30	93.98	91.31
Test Accuracy	95.97	93.22	91.28

demonstrates the robustness of our model, suggesting that it maintains performance even under challenging conditions.

We also evaluated our model by testing it with noisy test data while using clean data for training. The results revealed minimal differences in accuracy, indicating that the model remains resilient and performs well even when faced with noisy data during testing as shown in Table III. Hence, checking the model for both: 1) Evaluating noise adaptation: Train on noisy data and test on noisy data. 2) Robustness testing: Train on clean data and test on noisy data.

TABLE III
PERFORMANCE EVALUATION OF PROPOSED METHOD WHEN ONLY TEST DATA IS NOISY DATA.

	Std = 0.5
Train Accuracy	96.21
Test Accuracy	92.61

V. CONCLUSION

In this research, we propose a simple deep learning architecture that combines Hilbert Schmidt Information Criterion layers with linear layers trained via backpropagation. Our approach addresses several limitations of traditional BP training, such as the challenges posed by vanishing and exploding gradients, and the substantial memory overhead associated with deep networks. By integrating linear layers between HSIC layers, our architecture effectively leverages the benefits of local training and global optimization, resulting in improved model performance.

The experimental results across various datasets, including MNIST, CIFAR-10, and Fashion MNIST, demonstrate that our model outperforms the HSIC-only approach, particularly in handling noisy data. This indicates that the addition of linear layers enhances the network's ability to learn and retain crucial features, contributing to more accurate predictions. Furthermore, our architecture achieves comparable accuracy to existing models but with a more compact and memory-efficient design, highlighting its practical advantages in resource-constrained environments. Future research could delve deeper into developing training techniques that completely eliminate the need for backpropagation, potentially leading to more efficient and scalable neural network architectures.

REFERENCES

[1] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5, 2015.

[2] P. Czyż, F. Grabowski, J. Vogt, N. Beerenwinkel, and A. Marx, "Beyond normal: On the evaluation of mutual information estimators," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[3] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *CoRR*, vol. abs/1612.00410, 2016.

[4] W.-D. K. Ma, J. Lewis, and W. B. Kleijn, "The hsic bottleneck: Deep learning without back-propagation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 5085–5092, 2020.

[5] H. Khan, H. Liu, and C. Liu, "Missing label imputation through inception-based semi-supervised ensemble learning," *Advances in Computational Intelligence*, vol. 2, no. 1, p. 10, 2022.

[6] Z. Zhu, T. Liu, and Y. Liu, "A second-order approach to learning with instance-dependent label noise," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10113–10123, 2021.

[7] O. Albahri, A. Zaidan, A. Albahri, B. Zaidan, K. H. Abdulkareem, Z. Al-Qaysi, A. Alamoodi, A. Aleesa, M. Chyad, R. Alesa, *et al.*, "Systematic review of artificial intelligence techniques in the detection and classification of covid-19 medical images in terms of evaluation and benchmarking: Taxonomy analysis, challenges, future solutions and methodological aspects," *Journal of infection and public health*, vol. 13, no. 10, pp. 1381–1396, 2020.

[8] T. Nakatani, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *proc. INTERSPEECH*, vol. 2019, pp. 1408–1412, 2019.

[9] M. Nabipour, P. Nayyeri, H. Jabani, S. Shahab, and A. Mosavi, "Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis," *Ieee Access*, vol. 8, pp. 150199–150212, 2020.