Counter-Current Learning: A Biologically Plausible Dual Network Approach for Deep Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite its widespread use in neural networks, error backpropagation has faced criticism for its lack of biological plausibility, suffering from issues such as the backward locking problem and the weight transport problem. These limitations have motivated researchers to explore more biologically plausible learning algorithms that could potentially shed light on how biological neural systems adapt and learn. Inspired by the counter-current exchange mechanisms observed in biological systems, we propose counter-current learning (CCL), a biologically plausible framework for credit assignment in neural networks. This framework employs a feedforward network to process input data and a feedback network to process targets, with each network enhancing the other through anti-parallel signal propagation. By leveraging the more informative signals from the bottom layer of the feedback network to guide the updates of the top layer of the feedforward network and vice versa, CCL enables the simultaneous transformation of source inputs to target outputs and the dynamic mutual influence of these transformations. Experimental results on MNIST, FashionMNIST, CIFAR10, CIFAR100, and STL-10 datasets using multi-layer perceptrons and convolutional neural networks demonstrate that CCL achieves comparable performance to other biologically plausible algorithms while offering a more biologically realistic learning mechanism. Furthermore, we showcase the applicability of our approach to an autoencoder task, underscoring its potential for unsupervised representation learning. Our work presents a direction for biologically inspired and plausible learning algorithms, offering an alternative mechanisms of learning and adaptation in neural networks.

1 Introduction

In deep learning, *biological plausibility* refers to the properties that deep learning algorithms could respect to avoid inconsistency with current understandings of neural circuitry or violation of fundamental physical constraints, such as the localized nature of synaptic plasticity [Grossberg, 1987, Crick, 1989]. Consequently, error backpropagation (BP), despite its wide application, has been frequently criticized for its lack of biological plausibility, particularly for the following three challenges: (a) The *weight transport problem*, which arises because BP requires the feedback pathway to use the same set of weights as the feedforward process, a mechanism not observed in biological systems [Burbank and Kreiman, 2012, Bengio et al., 2015, Lillicrap et al., 2016]. (b) The *non-local credit assignment problem* arises because backpropagation relies on the global error signal to update the synaptic weights throughout the network, instead of depending on local errors derived from local loss computation.[1]. (c) The *backward locking problem* occurs because, in BP, each data sample must await the completion of both forward and backward computations of the previous sample, impeding online learning capabilities [Jaderberg et al., 2017, Czarnecki et al., 2017]. These limitations have propelled the development of alternative credit assignment methods that aim to better align with biological principles and address these significant issues [Lillicrap et al., 2016, Crafton et al., 2019, Launay et al., 2020, Nøkland, 2016, Bengio, 2014, Lee et al., 2015, Ororbia and Mali, 2019, Meulemans et al., 2020, 2021, Dellaferrera and Kreiman, 2022, Shibuya et al., 2023].

Reaching Biological Plausibility With a Dual Network Structure. To address the weight transport problem, we leverage a dual network architecture for processing feedback signals, which uses a different set of weights from the forward network. To tackle the non-local credit assignment issue, we use pairwise local loss, computing the difference in layerwise activations between the feedforward and feedback networks, and ensuring the local loss only updates local weight parameters through gradient detaching. We solve the backward-locking problem by preventing the feedback network from reusing latent activations and output signals from the feedforward networks. Since the feedback network operates independently of the output (prediction), the forward and feedback processes can occur simultaneously. These enhancements make our approach not only more biologically plausible but also potentially more effective in complex scenarios.

Analogy to Biological Counter-Current Mechanism. Our scheme draws inspiration from nature's *counter-current exchange mechanisms*, observed in fish gills, animal vessels, and renal systems. These physiological mechanisms use an anti-parallel structure to optimize resource or energy exchange between two flows. Similarly, our dual network learning scheme allows the input signals in the forward network, flowing from input space to target space, to receive target domain information from the target-to-source signal flow (in the feedback network) and reciprocally share their source information. Therefore, we name our learning scheme "counter-current learning," as this reciprocal exchange mirrors the efficiency and optimization seen in biological systems.

Contributions. This paper aims to introduce counter-current learning as a novel, biologically plausible alternative. We validate our approach through experiments on MNIST, FashionMNIST, CIFAR10, CIFAR100, and STL-10 using MLP or CNN architectures. Additionally, we demonstrate the effectiveness of our model in autoencoder-based tasks, which, to our knowledge, represents the first application of biologically plausible algorithms in this area. By effectively addressing the weight transposition, non-local credit assignment, and backward locking issues, the counter-current learning framework provides a promising avenue for advancing biological plausibility.

2 Literature Review

Target Propagation: Addressing Biological Plausibility. The target propagation (TP) family [Bengio, 2014] and its variants (e.g., local target representations) [Ororbia et al., 2018, 2023], first explored in the late 1980s [Le Cun, 1986, Le Cun and Fogelman-Soulié, 1987], have been developed to optimize the neural networks by using locally generated error signals. TP explicitly constructs local targets for each layer using a separate feedback network. Take difference target propagation (DTP) Lee et al. [2015] for example, an idealized global target signal is computed based on the labels and the prediction error at the output layer. Then, the local idealized targets are generated by (1) propagating the idealized global targets through the feedback network and (2) computing a linear correction using the activations from the forward network. Subsequently, local losses are computed by comparing the layer activations with their corresponding local targets. The weights of both the forward and feedback networks are updated based on these local losses. Notable variants such as Direct Difference Target Propagation (DDTP) [Meulemans et al., 2020], Local-Difference Reconstruction Loss (L-DRL) [Ernoult et al., 2022], and Fixed-Weight Target Propagation Shibuya et al. [2023] further refine this approach by introducing mechanisms to improve feedback weight training and enhance the accuracy of local error signals. Despite these advancements, TP methods still encounter the backward locking issue, since TP methods depend on the forward network's outputs and intermediate activations to compute targets. Moreover, recent iterations of TP algorithms, such as DDTP and L-DRL, can be computationally expensive. They require additional feedback

[1]In biological systems, synaptic plasticity is believed to be governed by local learning rules, such as Hebbian learning, where synaptic changes depend on the correlated activity of the pre-and post-synaptic neurons [Dan and Poo, 2004, Bartunov et al., 2018, Whittington and Bogacz, 2019]

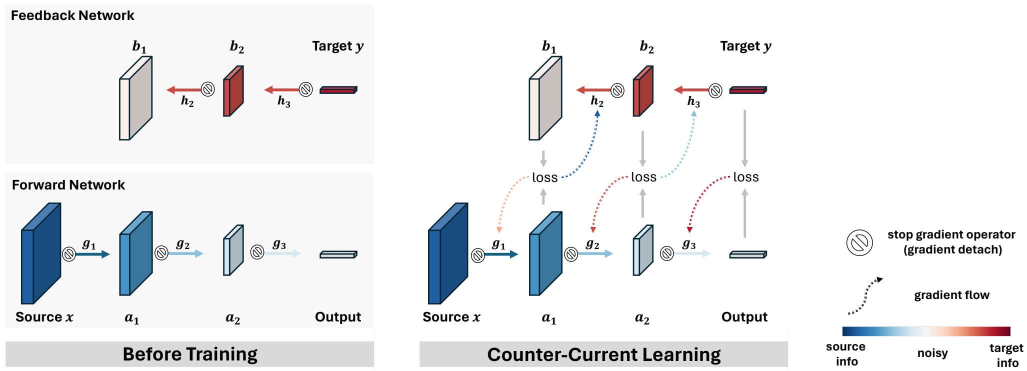

Figure 1: **Overview of the Counter-Current Learning Framework:** (a) **At initialization**, the counter-current learning framework establishes a dual network structure, with a forward network that maps the input to the target output, and a complementary feedback network that mirrors the forward network's architecture but propagates information in the opposite direction. Due to random weight initialization, the information content in both networks decreases from the bottom to the top layers. Notably, the dependency of the gradient on earlier layer parameters is interrupted using the gradient detachment operator. (b) **During training**, the losses are computed in a layer-wise manner, i.e., by calculating the difference of activations from corresponding layer pairs between the forward and feedback networks, allowing the networks to learn from each other's complementary information.

weight update loop per data batch, leading to a three to six-fold increase in training time compared to traditional backpropagation [Meulemans et al., 2020, Ernoult et al., 2022, Shibuya et al., 2023].

Other Efforts in Enhancing Biological Plausibility. In addition to TP, several other methods have been proposed to overcome the biological implausibility of traditional backpropagation. The feedback alignment (FA) [Lillicrap et al., 2016, Nøkland, 2016, Crafton et al., 2019, Launay et al., 2020, Refinetti et al., 2021] family uses random feedback weights, instead of the transpose of the weight in the feedforward layer, to approximate the error gradient, thereby eliminating the need for precise synaptic symmetry. To resolve the backward locking problem, direct random target projection (DRTP) [Frenkel et al., 2021] proposed to randomly project the target signals to each layer as ideal targets. While achieving biological plausibility, DRTP encounters a significant performance drop concerning BP compared to FA algorithms Frenkel et al. [2021], Dellaferrera and Kreiman [2022]. Block-local learning (BLL) Kappel et al. [2023] explores block-wise target signal propagation; however, the algorithm requires backpropagation to update layers in the same block.

3 Counter-Current Learning Framework

In this section, we present the counter-current learning (CCL) framework, as shown in Figure 1, focusing on its formulation and key components.

Setup and Feedforward Network. Consider input space $\mathcal{X}$ and output space $\mathcal{Y}$, each with dimensions d_0 and d_L, respectively. The objective is to learn a mapping $F : \mathcal{X} \to \mathcal{Y}$ that minimizes the discrepancy between the predicted output and the target. We adopt an L-layered feed-forward neural network with activation function σ. Let $g_l(\cdot)$ denote the operation at layer l, and define $F_{fw} = g_L \circ g_{L-1} \circ \ldots \circ g_1$. Each g_l is parameterized by weights U_l. The output of layer l is $a_l = g_l(a_{l-1}) = \sigma(U_l a_{l-1})$, where $a_0 = x$.

Feedback Network. The proposed learning scheme introduces a complementary backward function F_{fw} that mirrors F_{fw} in an anti-parallel manner. F_{bw} comprises layers $[h_L, \ldots, h_1]$, with each h_l parameterized by weights V_l. We define $F_{bw} = h_1 \circ \ldots \circ h_L$. The output of layer l is $b_{l-1} = h_l(b_l) = \sigma(V_l b_l)$, with $b_L = y$. The dimensions of hidden layers align between F_{fw} and F_{bw}.

Stop Gradient Operation. To address the backward locking problem and ensure local synaptic learning, we use the `SG()` operation to decouple activations from weights in previous layers, disrupting the long error-backpropagation chain into local update segments. The `SG()` can be implemented using PyTorch gradient detach operation easily. In CCL, each layer's input is processed with the

```python
from torch.nn import Linear, Module
import torch.nn.functional as F

class C2Model(Module):
    def __init__(self):
        super(C2Model, self).__init__()
        self.enc1 = C2Linear(784, 256)
        self.enc2 = C2Linear(256, 20)
        self.enc3 = C2Linear(20, 10)
    def fw_pass(self, x, detach):
        a1 = self.enc1.fw_pass(x, detach)
        a2 = self.enc2.fw_pass(a1, detach)
        a3 = self.enc3.fw_pass(a2, detach)
        return [x, a1, a2, a3]
    def bw_pass(self, target, detach):
        b2 = self.enc3.bw_pass(target, detach)
        b1 = self.enc2.bw_pass(b2, detach)
        b0 = self.enc1.bw_pass(b1, detach)
        return [b0, b1, b2, target]

class C2Linear(Module):
    def __init__(self, in_dims, out_dims):
        super(C2Linear, self).__init__()
        self.fw_layer = Linear(in_dims, out_dims)
        self.bw_layer = Linear(out_dims, in_dims)
    def fw_pass(self, x, detach):
        if detach: x = x.detach()
        return F.elu(self.fw_layer(x))
    def bw_pass(self, x, detach):
        if detach: x = x.detach()
        return F.elu(self.bw_layer(x))
```

```python
from torchvision.datasets import MNIST
from torch.utils.data import DataLoader
from nn.functional import one_hot
from torch import optim

def train_CCL_step(model, inputs, labels):
    fw_actvs = model.fw_pass(inputs, True)
    bw_actvs = model.bw_pass(labels, True)
    loss = 0
    for a, b in zip(fw_actvs, bw_actvs):
        loss += F.mse_loss(a, b)
    return loss

def train_BP_step(model, inputs, labels):
    fw_acts = model.fw_pass(inputs, False)
    return F.mse_loss(fw_acts[-1], labels)

train_dataset = MNIST(root='./data')
train_loader = DataLoader(train_dataset)
model = C2Model()
optimizer = optim.Adam(model.parameters())
for inputs, labels in dataloader:
    inputs = inputs.view(inputs.size(0), -1)
    # For CCL
    labels = one_hot(labels, 10).float()
    loss = train_CCL_step(model, inputs, labels)
    # For BackProp
    # loss = train_BP_step(model, inputs, labels)
    # For both CCL and BackProp
    loss.backward()
    optimizer.step()
```

Figure 2: Code Snippet For Counter-Current Learning With Dual Network Architecture.

SG() operation. To avoid confusion, we use the hat symbol to denote the exact activations in the CCL paradigm. Specifically, for $1 \leq l \leq L$:

$$\begin{aligned} \hat{a}_l &= \hat{g}_l(\hat{a}_{l-1}) = \sigma(U_l \mathtt{SG}(\hat{a}_{l-1})), \\ \hat{b}_{l-1} &= \hat{h}_l(\hat{b}_l) = \sigma(V_l \mathtt{SG}(\hat{b}_l)), \end{aligned} \tag{1}$$

where $\hat{a}_0 = x$ and $\hat{b}_L = y$.

Loss Objective Function. The objective of the counter-current learning algorithm is to minimize the difference between activations of F_{fw} and F_{bw} across all layers:

$$\min_{\theta} \sum_{l=0}^{L} \|\hat{a}_l - \hat{b}_l\|, \tag{2}$$

where $\theta = \{U_1, \ldots, U_L, V_1, \ldots, V_L\}$ are learnable parameters. For example, let us consider the local loss function at the first layer (i.e., $l = 1$), where the loss function is $\min_\theta \|\hat{a}_1 - \hat{b}_1\|$. Plugging Eq. 1 gives us $\min_{\{U_1, V_2\}} \|\sigma(U_1 \mathtt{SG}(\hat{a}_0)) - \sigma(V_2 \mathtt{SG}(\hat{b}_2))\|$, where $\mathtt{SG}(\hat{a}_0)$ and $\mathtt{SG}(\hat{b}_2)$ are treated as constants due to the stop gradient operation.

Biological Plausibility. We examine the biological plausibility of the proposed counter current learning scheme. This framework mitigates the weight transport problem by using a different weight parameterization for the feedback network. For the non-local credit assignment problem, the update of the parameters is driven by local loss, instead of the back-propagated global error signals. Finally, we address the backward update problem by removing the dependency of the backward network and the forward network with careful gradient detachment.

Implementation. In Figure 2, we present a code snippet for the CCL algorithm in PyTorch, tailored for an MNIST classification. It shows the independence of the forward process and the feedback (backward) process from each other. The main `C2Model` module comprises three `C2Linear` layers, each inherits from `nn.Module` and consists of `fw_pass` and `bw_pass` for forward propagation and backward propagation, respectively. These functions accept an additional Boolean input, `detach`, allowing for the quick toggling between non-local (i.e., BP) and local learning (i.e., CCL) modes. For loss computation, the `train_CCL_step` function calculates the loss for counter-current learning. Conversely, the `train_BP_step` function computes the loss for error backpropagation.

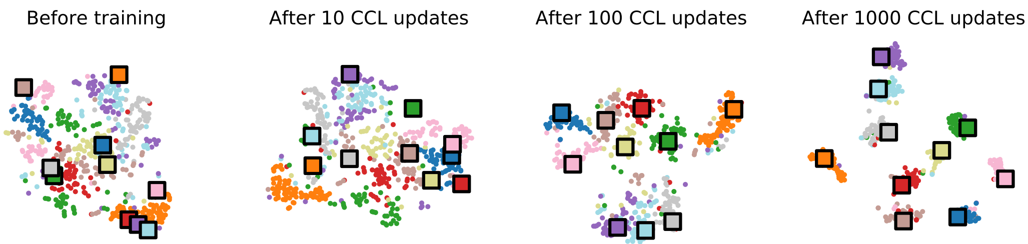

Figure 3: **Dynamic Feature Alignment Between Forward and Backward Models During Counter-Current Learning.** This series of t-SNE plots demonstrates the evolution of feature space alignment over different stages of training. Circular dots represent features from the forward network processing MNIST images, while squares depict features from the feedback network handling one-hot encoded labels. Each color represents a distinct class, with every subplot providing an independent t-SNE visualization. This emphasizes how distinct classes increasingly converge within and across the forward and backward models as training progresses, highlighting the dynamic and reciprocal nature of learning within the counter-current framework.

Table 1: Test performance on MNIST, FashionMNIST, CIFAR10, and CIFAR100, evaluated using multi-layer perceptrons. Performance metrics are reported for error backpropagation (BP), feedback alignment (FA), target propagation (DTP), DTP with difference reconstruction loss (DRL), local difference reconstruction loss (L-DRL), fixed-weight difference target propagation (FW-DTP), and cross-correlation loss (CCL). Best values per task are **bolded**, and second-best values are <u>underlined</u>.

	MNIST	FashionMNIST	CIFAR10	CIFAR100
BP [Rumelhart et al., 1986]	**98.19** $\pm$ 0.10	**89.58** $\pm$ 0.25	<u>50.03</u> $\pm$ 0.31	**22.55** $\pm$ 0.19
FA [Lillicrap et al., 2016]	96.96 $\pm$ 0.05	87.38 $\pm$ 0.12	45.76 $\pm$ 0.38	<u>22.13</u> $\pm$ 0.41
DTP Nøkland [2016]	97.27 $\pm$ 0.06	87.35 $\pm$ 0.99	42.86 $\pm$ 1.94	19.87 $\pm$ 1.50
DRL Meulemans et al. [2020]	93.05 $\pm$ 0.24	83.40 $\pm$ 0.18	42.09 $\pm$ 0.27	19.94 $\pm$ 0.28
L-DRL Ernoult et al. [2022]	93.29 $\pm$ 0.21	83.60 $\pm$ 0.20	42.19 $\pm$ 0.30	19.96 $\pm$ 0.27
FW-DTP Shibuya et al. [2023]	97.20 $\pm$ 0.16	87.78 $\pm$ 0.47	45.91 $\pm$ 0.60	21.09 $\pm$ 0.31
DRTP Frenkel et al. [2021]	92.16 $\pm$ 0.18	82.03 $\pm$ 0.56	33.85 $\pm$ 0.43	15.53 $\pm$ 0.33
CCL (ours)	<u>98.13</u> $\pm$ 0.10	<u>88.58</u> $\pm$ 0.29	**52.73** $\pm$ 0.59	21.76 $\pm$ 0.22

Code Availability. The code is available in the supplementary material.

4 Experiments

4.1 Counter Current Learning Facilitates Dual Network Feature Alignment

To investigate the alignment of latent features within the counter-current learning framework, we visualized embeddings from the penultimate layer of the forward network and the corresponding second layer of the feedback network at various stages of training. We employ a six-layer neural net trained on MNIST and analyze embeddings from both networks. t-SNE was applied independently to these embeddings at each training iteration to effectively visualize the evolution of feature spaces.

As illustrated in Figure 3, the embeddings from the forward model trained on MNIST data are represented by colored dots, while the embeddings from the backward model related to one-hot encoded labels are denoted by outlined squares of the same color. Throughout the training process, embeddings from the same class progressively align between the forward and backward models, suggesting that the forward and backward models mutually guide each other's feature representations toward a coherent and discriminative structure.

4.2 Classification Performance

Task Setup. We evaluate the performance of our proposed method against several biologically plausible algorithms, including direct target propagation (DTP), DTP with difference reconstruction loss

(DRL), local difference reconstruction loss (L-DRL), and fixed-weight difference target propagation (FW-DTP). The evaluation is conducted on MNIST, FashionMNIST, CIFAR-10, and CIFAR-100. All experiments are performed using stochastic gradient descent optimization with 100 epochs. We use cross-validation over 5 different random seeds and report the testing performance. The models are implemented using the PyTorch deep learning framework, and the code is available in the Supplementary Material. For the experiments on multi-layer perceptrons, we apply image normalization as a preprocessing step. For the convolutional neural network experiments, we use additional data augmentation techniques, including cropping and horizontal flipping. For the counter-current learning (CCL) algorithm, we search across different learning rates and gradient norm clipping values to find the optimal hyperparameters following cross-validation, as detailed in Appendix 6.1.

Multi-Layer Perceptrons (MLP). We follow Shibuya et al. [2023][2] for experimental setup and hyperparameter selection. For MNIST and FashionMNIST, we employ a fully connected network with 6 layers, each having 256 units. For CIFAR-10 and CIFAR-100, we use a fully connected network with 4 layers, each containing 1,024 units. We use the hyperbolic tangent activation function for BP and TP variant algorithms, while CCL adopts an ELU activation function. The results for MLPs are shown in Table 1, which demonstrates that the CCL obtains comparable results to error backpropagation and bears consistency with other biologically plausible algorithms.

Table 2: **Training Time Comparison on MNIST and CIFAR10.** The average running time of Error backpropagation (BP) serves as baselines, utilized for calculating the averaged training time ratio along with the standard deviation of the other algorithms. FA and DRTP are not shown for their algorithmic similarity with BP. Best values per task are **bolded**, second best are <u>underlined</u>.

ARCHITECUTURE	MNIST 6 LAYERS	CIFAR10 4 LAYERS
BP	1.00	1.00
DTP	4.40 $\pm$1.31	2.86 $\pm$0.41
DRL	9.63 $\pm$2.93	4.56 $\pm$0.69
L-DRL	7.74 $\pm$2.35	4.79 $\pm$1.02
FWDTP-BN	<u>2.92</u> $\pm$0.83	<u>1.99</u> $\pm$0.50
CCL (OURS)	**1.14** $\pm$0.83	**1.12** $\pm$0.50

Table 3: **Test Accuracy on CIFAR10, CIFAR100, and STL10 Using Convolutional Neural Network.** The metrics are reported for error backpropagation (BP) and cross-correlation loss (CCL).

	CIFAR10	CIFAR100	STL10
BP	87.12 $\pm$1.76	51.92 $\pm$0.48	51.27 $\pm$1.90
CCL (OURS)	82.94 $\pm$0.53	56.29 $\pm$0.25	45.28 $\pm$2.58

MLP Runtime Analysis. To assess the runtime efficiency of the algorithms, we compare the training time across MLP models and datasets (Table 2). The results show that counter-current learning (CCL) consistently outperforms the TP family algorithms in terms of training time. Note that the forward and feedback process for CCL is performed sequentially in this experiment.

Convolutional Neural Network. We also evaluate CCL on convolutional neural networks (CNN) consisting of five convolutional layers (each with a kernel size of 3 and a max-pooling operation with a kernel size of 2) followed by a linear classifier, tested on CIFAR-10, CIFAR-100, and STL-10. As shown in Table 3, our CCL-based model performs comparably to, or slightly inferiorly than, the BP-based model. Additionally, we visualize the kernels in the first convolutional layer to inspect the learned representations in Figure 4, demonstrating that CCL enables the model to learn meaningful convolutional kernels without using error backpropagation. Furthermore, we compare our CNN results on CIFAR-10 with those of L-DRL Ernoult et al. [2022] in Appendix 6.2, while FW-DTP Shibuya et al. [2023] does not include CNN implementations.

[2]Codebase: `https://github.com/TatsukichiShibuya/Fixed-Weight-Difference-Target-Propagation/tree/main`

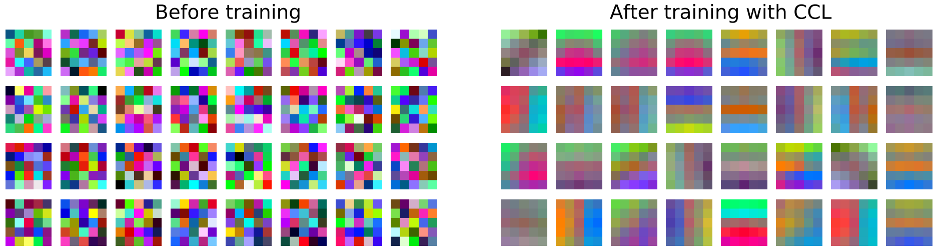

Figure 4: **Visualization of First Layer Convolutional Kernels of the Forward Model Trained on CIFAR-10 Using Counter-Current Learning.** The left subplot shows the randomly initialized kernels and the right subplot shows the kernels learned after training for 10 epochs.

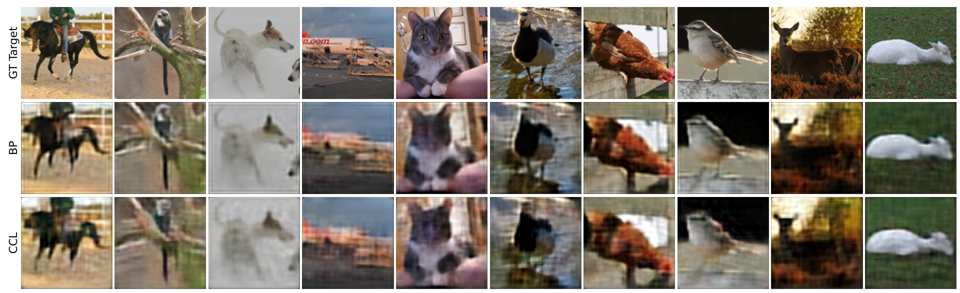

Figure 5: **Qualitative Comparison of an Eight-Layered Convolutional Autoencoder Trained Using Error Backpropagation (BP) and Counter-Current Learning (CCL).** The network structure does not contain skip connections. Testing set reconstruction results highlight CCL's comparable reconstruction as BP while achieving biological plausibility.

4.3 Auto-Encoder Performance

We explore the applicability of the counter-current learning (CCL) algorithm to autoencoders.

Auto-Encoder on STL-10 Dataset. A convolutional autoencoder with a four-layer encoder and a four-layer decoder is used. Different from the classification tasks, this architecture replaces the 2x2 kernel max-pooling with convolution layers with a stride of 2. Batch normalization is applied following each linear projection and before activation functions to ensure both stability and optimal performance. The hidden layers of the network are structured with dimensions of $[128, 256, 1024, 2048]$. Orthogonal weight initialization is used for training stability. Data augmentation techniques such as random cropping and horizontal flipping are incorporated. Hyperparameters including gradient clipping, learning rate, momentum, and weight decay are subjected to grid search, while cross-validation across five different seeds is employed to assess the reconstruction L2 loss.

Results. The test set's reconstruction metric—mean square error—is quantified as 0.0059 ± 0.0001 for BP and 0.0086 ± 0.0001 for CCL. The outcomes, illustrated in Figure 5, underscore the models' proficiency on the test set. While both BP and CCL adeptly capture the general image structure, occasional artifact introduction, such as blurring, is observed in CCL compared to BP. This suggests that while CCL augments certain facets of autoencoder training, further refinement or architectural adjustments may be imperative to minimize visual artifacts and enhance detail preservation.

4.4 Empirical Analysis of Learning Dynamics for Counter-Current Learning

In this section, we provide insights into the functioning of the proposed Counter-Current Learning (CCL) algorithm by examining the representation similarity between the forward and feedback networks. We start by analyzing the feature similarity between randomly initialized feedforward and feedback models. Following this, we focus on the feature alignment in the high-level feature regime. Our investigation reveals the emergence of a reciprocal learning structure, where the top layers of both networks benefit from the bottom layers of each other during training.

We trained a model on the MNIST classification task using a configuration of five convolutional layers topped with a linear classification head. The training was conducted over 160 steps with a

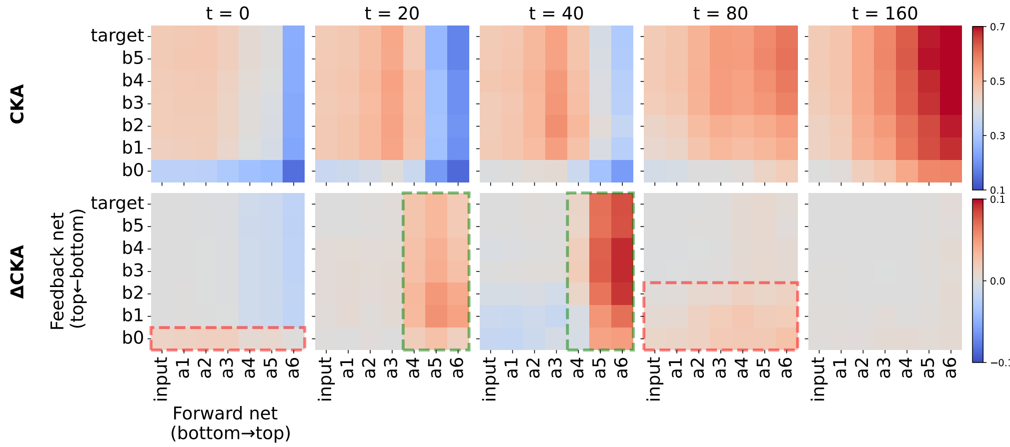

Figure 6: **Counter-Current Signal Propagation Enables Learning Through Reciprocal Representation Alignment. (Top) Centered Kernel Alignment (CKA)** between the forward and feedback networks during training. At the initial training step ($t = 0$), cross-network CKA is minimal, suggesting a low similarity between networks. As training progresses, CKA significantly increases, especially in the top layers of both networks, indicating high similarity in learned high-level representations. **(Bottom) Changes in CKA** between consecutive training steps (i.e., from step t to step $t + 1$) reveal significant increases in the top layers of both networks, consistent with our counter-current learning insights. Notably, increases are concentrated in the a_4, a_5, a_6 columns of the forward network and in the b_0, b_1, b_2 rows of the feedback network, as highlighted by the green dotted box. These changes align with the expected reciprocal and complementary learning dynamics.

batch size of 32, achieving an average testing accuracy of 88.88%. To measure the cross-network representational similarity, we utilized Centered Kernel Alignment (CKA) Kornblith et al. [2019], a metric known for its robustness to invertible linear transformations. This was applied to evaluate layer and architecture similarity. The results, obtained using five different seeds, are presented as averaged CKA values on the test set. Figure 6 displays the horizontal axis marking the activations of forward layers, ranging from the input MNIST images to the logits (i.e., a_6), whereas the vertical axis denotes the activations of the feedback network starting from one-hot labels (i.e., target).

Observation 1: Initialization Shows Noisy Top Layers and Misalignment Between Networks. At initialization, our premise that the bottom layers contain more relevant information is validated. The initial CKA ($t = 0$, top-left subplot in Figure 6) reveals low similarity between the a_6 column (i.e., the top layer of the forward network) and the feedback network. Similarly, the b_0 row (i.e., the top layer of the feedback network) shows low CKA values with the forward layers, confirming our hypothesis of feature misalignment at random initialization.

Observation 2: Alignment of High-Level Features During Training. As training progresses, high-level features (i.e., the top layers of the forward network and the bottom layers of the feedback network) begin to show increasing CKA values (Figure 6, $t = 20$ to $t = 160$, top row). This trend suggests that the forward and feedback networks gradually align their high-level representations.

Observation 3: Emergence of a Counter-Current Reciprocal Structure. The dynamics of the counter-current learning algorithm reveal significant changes in CKA, particularly at higher layers (illustrated in the bottom subplots of Figure 6). Notably, the increases are concentrated in the a_4, a_5, a_6 columns of the forward network and the b_0, b_1, b_2 rows of the feedback network. This pattern supports the counter-current intuition that top layers benefit from the more informative bottom layers, fostering a reciprocal learning structure that guides each network's optimization process using the most informative features available.

This empirical analysis underscores the hypothesis that the counter-current learning scheme effectively leverages complementary and reciprocal alignment of representations between the forward and feedback networks. By exploiting the informative features from the bottom layers of one network to refine the noisy features in the top layers of the other, the counter-current signal propagation algorithm achieves a biologically plausible and efficient learning mechanism.

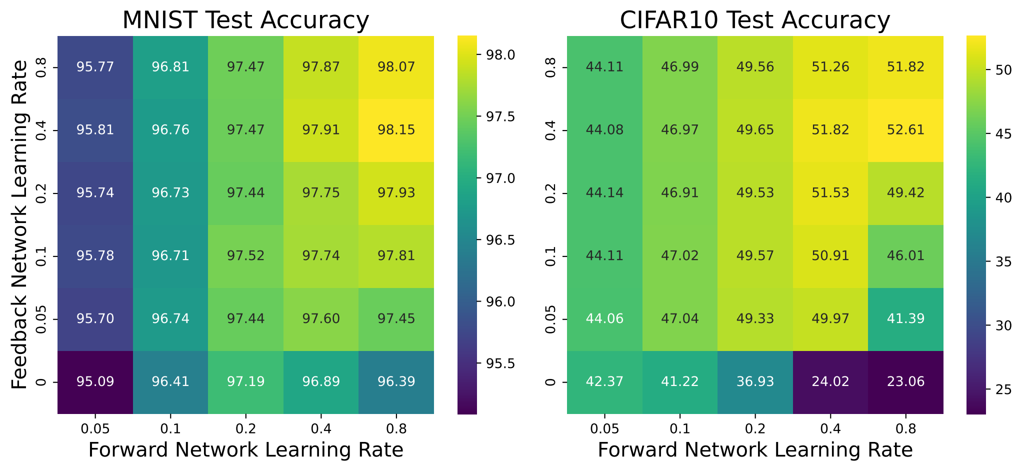

Figure 7: **Learning With Asymmetric Learning Rates in the Networks.** We investigate the influence of asymmetric learning rate in the forward and backward MLPs. This study demonstrates that effective learning can occur with asymmetric learning rates, even when the feedback network has a fixed random configuration (i.e., the learning rate for the feedback net is zero).

4.5 Ablation on Asymmetric Learning Rates for Dual Network Optimization

We delve deeper into the effects of varying learning rates between forward and feedback networks in CCL. As illustrated in Figure 7, our experiments confirm that asymmetric learning rates in forward and feedback networks facilitate effective and robust learning. Particularly noteworthy is our observation that robust learning outcomes are achievable even when the feedback network operates under a fixed random setting—specifically, with a zero learning rate (refer to the bottom rows of the figure). This suggests that the random projection of target labels by the feedback network conveys meaningful target domain information, echoing findings from the DRTP [Frenkel et al., 2021]. Moreover, our experiments indicate superior performance compared to the DRTP approach (refer to Table 1), possibly hinting that a random, non-linear neural network projection (i.e., CCL with a feedback learning rate of zero) is more beneficial than a mere random linear projection (i.e., DRTP) of labels. This could be due to the neural network's ability to maintain and leverage more inductive biases, which can be crucial for the hierarchical learning processes.

5 Conclusion

In this paper, we introduced the counter-current learning framework, a novel approach addressing the critical limitations of traditional error backpropagation. Our dual network architecture enables a dynamic and reciprocal interaction between feedforward and feedback pathways, supporting local learning and effectively resolving the backward locking problem. Our learning framework is validated across a diverse set of datasets including MNIST, FashionMNIST, CIFAR10, CIFAR100, and STL10, and in autoencoder tasks, demonstrates comparable performance with existing learning methods without compromising learning speed. This underscores the potential of our model to efficiently handle complex neural tasks and highlights its suitability for broader applications.

Limitations and Future Directions. We acknowledge several limitations that can guide future research in this field. Firstly, there is a need for further theoretical insights into the counter-current learning scheme, focusing on its learning dynamics, stability analysis, and inductive biases. Secondly, continuous exploration of this dual model architecture is essential, such as integrating residual connections or self-attention modules. Thirdly, hardware acceleration to streamline forward-feedback computation through parallelization could potentially reduce computation time and yield improved results within the same time frame. Lastly, exploring the integration of counter-current learning with other biologically plausible alternatives holds promise for advancing research in this domain.

References

Sergey Bartunov, Adam Santoro, Blake Richards, Luke Marris, Geoffrey E Hinton, and Timothy Lillicrap. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. *Advances in neural information processing systems*, 31, 2018.

Yoshua Bengio. How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906*, 2014.

Yoshua Bengio, Dong-Hyun Lee, Jorg Bornschein, Thomas Mesnard, and Zhouhan Lin. Towards biologically plausible deep learning. *arXiv preprint arXiv:1502.04156*, 2015.

Kendra S Burbank and Gabriel Kreiman. Depression-biased reverse plasticity rule is required for stable learning at top-down connections. *PLoS computational biology*, 8(3):e1002393, 2012.

Brian Crafton, Abhinav Parihar, Evan Gebhardt, and Arijit Raychowdhury. Direct feedback alignment with sparse connections for local learning. *Frontiers in neuroscience*, 13:450947, 2019.

Francis Crick. The recent excitement about neural networks. *Nature*, 337(6203):129–132, 1989.

Wojciech Marian Czarnecki, Grzegorz Świrszcz, Max Jaderberg, Simon Osindero, Oriol Vinyals, and Koray Kavukcuoglu. Understanding synthetic gradients and decoupled neural interfaces. In *International Conference on Machine Learning*, pages 904–912. PMLR, 2017.

Yang Dan and Mu-ming Poo. Spike timing-dependent plasticity of neural circuits. *Neuron*, 44(1): 23–30, 2004.

Giorgia Dellaferrera and Gabriel Kreiman. Error-driven input modulation: Solving the credit assignment problem without a backward pass. In *International Conference on Machine Learning*, pages 4937–4955. PMLR, 2022.

Maxence M Ernoult, Fabrice Normandin, Abhinav Moudgil, Sean Spinney, Eugene Belilovsky, Irina Rish, Blake Richards, and Yoshua Bengio. Towards scaling difference target propagation by learning backprop targets. In *International Conference on Machine Learning*, pages 5968–5987. PMLR, 2022.

Charlotte Frenkel, Martin Lefebvre, and David Bol. Learning without feedback: Fixed random learning signals allow for feedforward training of deep neural networks. *Frontiers in neuroscience*, 15:629892, 2021.

Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.

Takashi Ishida, Ikko Yamane, Tomoya Sakai, Gang Niu, and Masashi Sugiyama. Do we need zero training loss after achieving zero training error? In *International Conference on Machine Learning*, pages 4604–4614. PMLR, 2020.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *International conference on machine learning*, pages 1627–1635. PMLR, 2017.

David Kappel, Khaleelulla Khan Nazeer, Cabrel Teguemne Fokam, Christian Mayr, and Anand Subramoney. Block-local learning with probabilistic latent representations. *arXiv preprint arXiv:2305.14974*, 2023.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

Julien Launay, Iacopo Poli, François Boniface, and Florent Krzakala. Direct feedback alignment scales to modern deep learning tasks and architectures. *Advances in neural information processing systems*, 33:9346–9360, 2020.

Yann Le Cun. Learning process in an asymmetric threshold network. In *Disordered systems and biological organization*, pages 233–240. Springer, 1986.

Yann Le Cun and Françoise Fogelman-Soulié. Modèles connexionnistes de l'apprentissage. *Intellectica*, 2(1):114–143, 1987.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part I 15*, pages 498–515. Springer, 2015.

Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7(1): 13276, 2016.

Alexander Meulemans, Francesco Carzaniga, Johan Suykens, João Sacramento, and Benjamin F Grewe. A theoretical framework for target propagation. *Advances in Neural Information Processing Systems*, 33:20024–20036, 2020.

Alexander Meulemans, Matilde Tristany Farinha, Javier García Ordóñez, Pau Vilimelis Aceituno, João Sacramento, and Benjamin F Grewe. Credit assignment in neural networks through deep feedback control. *Advances in Neural Information Processing Systems*, 34:4674–4687, 2021.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Alexander G Ororbia and Ankur Mali. Biologically motivated algorithms for propagating local target representations. In *Proceedings of the aaai conference on artificial intelligence*, pages 4651–4658, 2019.

Alexander G Ororbia, Ankur Mali, Daniel Kifer, and C Lee Giles. Conducting credit assignment by aligning local representations. *arXiv preprint arXiv:1803.01834*, 2018.

Alexander G Ororbia, Ankur Mali, Daniel Kifer, and C Lee Giles. Backpropagation-free deep learning with recursive local representation alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9327–9335, 2023.

Maria Refinetti, Stéphane d'Ascoli, Ruben Ohana, and Sebastian Goldt. Align, then memorise: the dynamics of learning with feedback alignment. In *International Conference on Machine Learning*, pages 8925–8935. PMLR, 2021.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Tatsukichi Shibuya, Nakamasa Inoue, Rei Kawakami, and Ikuro Sato. Fixed-weight difference target propagation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9811–9819, 2023.

James CR Whittington and Rafal Bogacz. Theories of error back-propagation in the brain. *Trends in cognitive sciences*, 23(3):235–250, 2019.

Hongwei Yong, Jianqiang Huang, Xiansheng Hua, and Lei Zhang. Gradient centralization: A new optimization technique for deep neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 635–652. Springer, 2020.

6 Appendix

6.1 Experimental Setup

Hyperparameter Search. For experiments with MLP architectures, we conduct hyperparameter searches for each algorithm. We run all the combinations of the hyperparameters with 5 different random seeds and then select the hyperparameter set with the highest accuracies (or lowest loss for auto-encoder task) evaluated on the validation set and test on the testing set. For DTP, DRL, L-DRL, and FWDTP, we search across forward learning rates $[0.3, 1, 3]$, step sizes $[0.001, 0.003, 0.01, 0.03, 0.1]$, and backward learning rates $[0.0001, 0.0003, 0.001, 0.003, 0.01]$. Note that FWDTP-BN does not have the backward learning rate hyperparameter. For DRTP, the search space includes learning rates $[0.010.030.10.3]$ and mean ($[0.0.05]$) and standard deviation ($[0.010.030.10.3]$) for random project matrix. For BP and FA, we use learning rates $[0.4, 0.2, 0.1, 0.05, 0.02, 0.01]$ for hyperparameter search. and gradient clip values of $[0.5, 1]$. For CCL, the search space includes learning rates $[0.2, 0.5, 1, 1.5, 2]$ and gradient clip values of $[0.5, 1]$.

Implementation Details. Training MLP and CNN models with CCL can be unstable initially since both the forward and feedback networks are randomly initialized. We use learning rate warm-up for the initial 200 steps for CCL, which is adopted in Ernoult et al. [2022]. Moreover, unlike algorithms in the target propagation family [Meulemans et al., 2020, Ernoult et al., 2022], where the feedback network weights are trained using additional for-loops to tune the weights for each data batch, we introduce some techniques to stabilize the training course. We found that normalizing the activations during loss computation helps stabilize the training process. Additionally, as counter-current learning can also suffer from feature collapse, where a trivial solution to all pairwise losses is to produce constant activations, we introduce a remedy. Inspired by layer-wise training, for each latent activation $X \in \mathbb{R}^{b \times d}$, where b stands for batch size and d stands for feature dimension, we added an additional L2 loss to minimize the difference between $\text{norm}(X)\text{norm}(X)^\top$ and the identity matrix. Finally, in contrast to error backpropagation, where the error signals can be zero and the weight updates can be small at the end of training, the layer-wise losses in CCL are seldom zero, thus the weights can keep changing during the training. This can lead to worse minima. To accommodate this, we use gradient centralization [Yong et al., 2020] to centralize the gradient for parameters for both BP and CCL and flooding method [Ishida et al., 2020] to prevent some weights from updating if the sample-wise difference between output and target is smaller than 0.2 in CCL on CNN.

6.2 Comparison of Implementation

Table 4: **Comparison of Results on CIFAR10 Using VGG-like Convolutional Neural Network.** * indicates that we report the results from the literature. .

	OUR IMPLEMENTATION	L-DRL ERNOULT ET AL. [2022]
TRAIN ON VALIDATION SET	×	✓
BP	87.12 ± 1.76	$89.07 \pm 0.22^*$
L-DRL ERNOULT ET AL. [2022]	-	$89.38 \pm 0.20^*$
CCL (OURS)	82.94 ± 0.53	-

We compare the results on CIFAR-10 using a VGG-like network architecture. As shown in Table 4, the results indicate that L-DRL [Ernoult et al., 2022] achieved similar testing accuracies to BP, reaching 89%. However, there are some issues with their implementation, as discussed in their GitHub repository. Specifically, they trained the models on the full training set, and the hyperparameter search space and the implementation of cross-validation were not disclosed in their paper.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: [Yes] The claim is supported by our extensive experiments using MLPs and CNNs.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: [Yes] We discuss thoroughly the limitation and future work in the conclusion section

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: [Yes] We specify the hyperparameter search space and use five random seeds for all main experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: [Yes] The data is publicly available and the code is included in supplement.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

Justification: [Yes] We use the default train-test split, specify the hyperparameter search space and use cross-validation to choose them, and use vanilla SGD with momentum.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: [Yes] Mean with standard deviation are shown.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: [Yes] We reveal the running time for different algorithms.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: [Yes] Anonymity is ensured.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: [NA] This paper is about a new learning scheme, as an alternative to back propagation. The algorithm itself does not have societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: [Yes] The asset is cited and its original link is shown. The authors do not specify their license on Github.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: [Yes] Please check the README.md in the supplementary material.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.