Conic Linear Units: Improved Model Fusion and Rotational-Symmetric Generative Model

Changqing Fu, Laurent D. Cohen

Conic Linear Units: Improved Model Fusion and Rotational-Symmetric Generative Model

Changqing Fu[1][a] and Laurent D. Cohen[1][b]
[1] *CEREMADE, University Paris Dauphine, PSL Research University, UMR CNRS 7534, Paris, 75016, France*
{*cfu, cohen*}*@ceremade.dauphine.fr*

Keywords: Generative AI, Deep Learning for Visual Understanding, Machine Learning Technologies for Vision

Abstract: We introduce Conic Linear Unit (CoLU), a natural generalization of commonly used activation functions in neural networks. The common pointwise ReLU activation is a projection onto the positive cone and is permutation symmetric. We propose a nonlinearity that goes beyond this symmetry: CoLU is a skew projection onto a hypercone towards the cone's axis. Due to the nature of this projection, CoLU enforces symmetry in a neural network with width C from the finite-order permutation group $\mathcal{S}(C)$ to the infinite-order rotation/reflection group $O(C-1)$, thus producing deep features that are motivated by the HSV color representation. Recent results on merging independent neural networks via permutation modulus can be relaxed and generalized to soft alignment modulo an optimal transport plan (Singh and Jaggi, 2020), which is useful in aligning models of different widths. CoLU aims to further alleviate the apparent deficiency of soft alignment. Our simulation indicates that CoLU outperforms existing generative models including Autoencoder and Latent Diffusion Model on small or large-scale image datasets. Additionally, CoLU does not increase the number of parameters and requires negligible additional computation overhead. The CoLU concept is quite general and can be plugged into various neural network architectures. Ablation studies on extensions to soft projections, general L^p cones, and the non-convex double-cone cases are briefly discussed.

1 INTRODUCTION

Scaling up neural networks is one thing, while reducing their redundancies is quite another. Aligning/Fusing different models into a canonical form (Ashmore and Gashler, 2015) is a useful way to simplify the model structure and reduce its redundancy. In the mean time, practically speaking, alignment enables different models to collaborate with each other. The term alignment in this context refers to fixing a base model and transforming any alternative ones so that the alternative model is very similar, if not equivalent, to the base model. This similarity, in our setting, is defined as the fact that both models can be linearly interpolated without losing much performance. In the generative model case, the effect of alignment is illustrated in figure 1, showing that the weights of aligned models can be linearly interpolated to obtain a model which performs well. In other words, the symmetry of the model's function space (visualized by the symmetry of the triangle in figure 1) is represented by symmetry group P (permutations), and we aim to let the quotient space (defined in section 2) modulo P be regular, in the sense that we hope the aligned models to be close to the base model, so that they can be merged via linear interpolation.

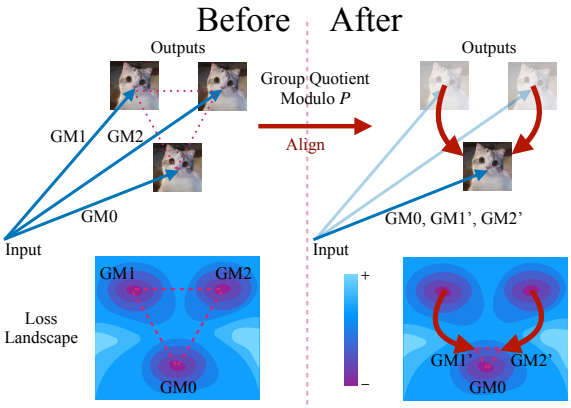

Figure 1: **Model Fusion.** Fixing a base generative model GM0, alternative models GM1, GM2 at different local minima on the loss landscape are aligned as GM1'=Align(GM1), GM2'=Align(GM2). Before the alignment, the linear interpolation between GM0 and GM1/GM2 has increased loss, whereas after the alignment, this increase is reduced.

[a] https://orcid.org/0000-0002-4485-9824
[b] https://orcid.org/0000-0002-3940-645X

However, existing alignment methods are often rough due to the narrow function space of the network's nonlinearities. For example, for the widely-used pointwise activation functions, the equivariance is associated with permutations. We propose a better design which enables smoother fusion. Meanwhile, it is intriguing that the new structure outperforms existing ones under certain validation criteria for various generative tasks.

1.1 Model Fusion

First of all, why do we need model fusion? We list several aspects from both applied and theoretical perspectives.

Practical Uses First, in the **federated learning** setting when the training task is distributed to local agents, fusion is a way to aggregate the parallelized training subtasks and synchronize the local models to obtain a global result (Wang et al., 2020a). Second, in the **knowledge distillation** setting when there is *no* access to training data such as for safety reasons, fusion makes it possible to directly ensemble models from *multiple* teacher models as an alternative to aggregating the training data. Third, in the **transfer learning** setting, the expensive computation overhead of re-training from scratch is saved by utilizing the information in *multiple* pre-trained models.

Theoretical Benefits First, **geometry**: associated with the correspondence between the aligned models is the symmetry of the network architecture's function space: the intuitive exchangeability of the hidden feature maps is algebraically characterized by the permutation modulus. In other words, CNN is *equivariant* under channel permutation group. Different forms of group equivariance has led to network designs with better efficiency and generalization ability. Second, **optimization**: learning the parameters of a neural network is a non-convex optimization problem. Alignment reshapes the loss landscape and largely *convexifies* it. Consequently, the training process largely converges to a minimum which is unique up to a transformation group. Third, **probability**: a fully-connected (or convolutional) layer is a linear ensembling among neurons (or feature maps) of previous layers. Fixing a certain set of trained neurons as the key's dictionary, alignment is used to *de-anonymize* an arbitrary set of neurons so that the activating behavior of an individual neuron at the same index follows the same pattern. The de-anonymization is deterministic, meaning each neuron is *assigned* an unrepeated key with probability 1.

1.2 Probabilistic Fusion

Then a natural question arises: what if the widths of the models to be aligned differ from each other? It's evident that deterministic assignment is not feasible when fusing models of varying width. In this unbalanced case, the assignment constraint can be relaxed by optimal transport. In a probabilistic sense, each key (channel index) is not deterministically assigned to a neuron, but instead a fuzzy mixture of neurons, whose probability values sum up to 1 (Singh and Jaggi, 2020). However, the model formed by the new set of neurons, each as a multi-identity mixture, does not behave the same way as in the unaligned model, since activating each individual neuron is no longer feasible. To resolve this apparent deficiency naturally caused by the relaxation, re-designing a more symmetric function is a must.

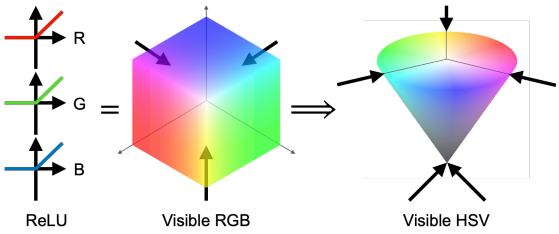

Figure 2: **Conic Linear Unit (three-channel case). Left:** ReLU. **Middle:** equivalent projective illustration of ReLU. **Right:** projective illustration of CoLU. With axes denoted as R, G, B, a 3D vector inside the positive cone represents a RGB color, and is visualized in the 3D space. Any points outside the cone is not visualized since the vector does not represent a color. The arrows point from the input to the output of the activation function.

Conic Linear Unit: From RGB to HSV The proposed activation is termed as **Conic Rectified Linear Unit** (abbreviated as CoReLU or **CoLU** for short), named after Rectified Linear Unit (ReLU). The pointwise ReLU is generalized to pixel-wise CoLU in the following way.

In the special case of three channels, CoLU is naturally motivated by switching from Red-Green-Blue (RGB) to the Hue-Saturation-Value (HSV) color representation, visualized in figure 2. ReLU is attending to perceptible color defined by the positive octant, where color components which are too dark (negative-valued) are not perceivable (mapped to zero). Instead of ReLU which preserves positive values, CoLU preserves low values on saturation, which means saturation values larger than a threshold value (proportional to the luminosity value) are mapped to the maximal threshold.

In high dimensions as is shown in figure 3, where the token space does not represent a color, migration

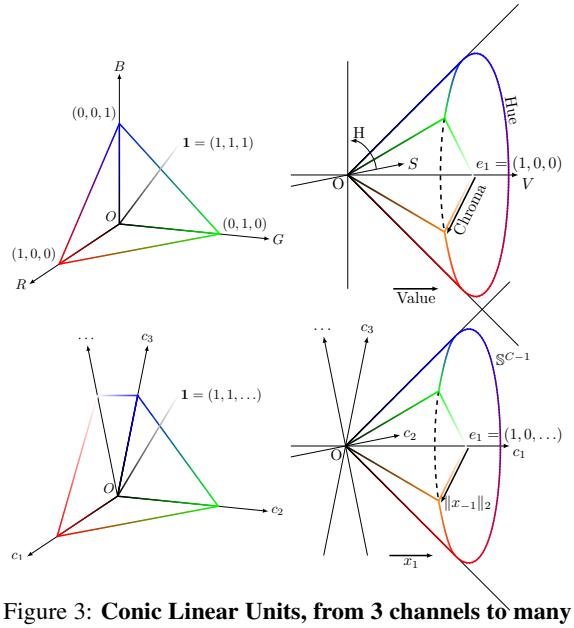

Figure 3: **Conic Linear Units, from 3 channels to many channels. Top:** cartesian axes of positive cones of ReLU and CoLU in three dimensions. **Bottom:** generalizations to high-dimensional space.

from the positive hypercone to the round hypercone motivates CoLU in hidden layers. Closed-form formula of CoLU is detailed in section 3.

Contributions We propose CoLU, a class of activation functions which present convolutional neural networks with rotational symmetry in the channel dimension. In principle, CoLU achieves improved soft alignment using optimal transport which is useful in federated learning. Intriguingly, CoLU outperforms baseline generative models on several tasks.

Organization of Sections After introducing preliminaries in section 2, we define CoLU in section 3, and describe the fusion algorithm in section 4. Quantitative improvements of conic activation and high-resolution generation results are presented in section 5.1. Then, weight alignment for both recognition and generative models are experimented in section 5.2. Related works are discussed in section 6 and the paper is concluded in section 7.

2 PRELIMINARIES

Consider a neural network learned on paired data (x_0, y), where the input and output are sampled from random variables following some data distribution $(x_0, y) \sim \mu$. The network is parameterized with weight $\mathbf{W} = \{\mathbf{W}_\ell\}_{\ell=1}^T$ trained from initialization $\mathbf{W} \sim \pi$, and the layer outputs are a sequence of hidden states $\{x(\ell)\}_{\ell=0}^T$, with terminal layers being the input $x(0) = x_0$ and output $x(T) = y$ respectively.

Definition 1 (Equivariance)**.** *A group P's action on a space M is defined by $P \times M \to M$, $(\mathbf{P}, x) \mapsto \mathbf{P}x$, $\forall \mathbf{P} \in P, x \in M$. A function $f : M \to M$ is said to admit the group P, or being equivariant under group action P if and only if the function commutes with the left multiplication as $f \circ \mathbf{P} = \mathbf{P} \circ f, \forall \mathbf{P} \in P$. Here $\mathbf{P}$ is called an alignment.*

Definition 2 (Hidden State Alignment)**.** *Two hidden states $x^{(0)}(\ell)$ and $x^{(1)}(\ell)$ are equivalent up to an alignment if and only if there exists an alignment $\mathbf{P}$ such that $x^{(1)}(\ell) = \mathbf{P}x^{(0)}(\ell)$, denoted as a relation $x^{(0)}(\ell) \sim_P x^{(1)}(\ell)$. Two neural networks are equivalent if and only if all hidden states are equivalent, that is $\forall \ell = 1, \ldots, T-1$, $x^{(0)}(\ell) \sim_P x^{(1)}(\ell)$, denoted as $x^{(0)} \sim_P x^{(1)}$.*

Example 1 (CNN with Pointwise Activation)**.** *The CNN's symmetry group P is channel permutations. A CNN (without skip connections) is defined by*

$$x(\ell+1) = \mathbf{W}_\ell \star \lambda(x(\ell)) \tag{1}$$

where λ is a pointwise activation such as ReLU, and the convolution is defined as $(w \star x)(\sigma, \omega_1, \omega_2) = \sum_{\sigma'=1}^{C'} \sum_{(\omega_1', \omega_2') \in \Omega'} x(\sigma', \omega_1 + \omega_1', \omega_2 + \omega_2') w(\sigma, \sigma', \omega_1', \omega_2')$ where Ω' is a small convolution window. The channel permutation group is defined as $P = S(C) := \{\mathbf{P}_\ell \in \mathbb{R}^{C \times C} : \exists$ permutation $\sigma, (\mathbf{P}_\ell)_{ij} = \mathbb{1}_{\{j=\sigma(i)\}} \forall i = 1, \ldots, C\}$.

Proof. Pointwise activation function commutes with permutation as $\mathbf{P}\lambda(x) = \lambda(\mathbf{P}x)$, and since the alignment is along the channel axis, by exchanging the order of multiplication and sum in the definition of convolution, we obtain $(\mathbf{PW}) \star x = \mathbf{W} \star (\mathbf{P}x)$. Therefore $x \mapsto \mathbf{W}_\ell \star \lambda(x)$ and $\mathbf{P}$ commutes. □

Theorem 1 (Weight Alignment)**.** *The CNN's hidden state alignment is equivalent to weight alignment. The weight alignment is defined as $\mathbf{W}_\ell^{(1)} = \mathbf{P}_\ell \mathbf{W}_\ell^{(0)} \mathbf{P}_{\ell-1}^{-1}, \forall \ell = 1, \ldots, T-1$, and denoted as $\mathbf{W}_\ell^{(0)} \sim_P \mathbf{W}_\ell^{(1)}$.*

Proof. Fix a base neural network with states $x^{(0)}(\ell)$ and parameters $\mathbf{W}_\ell^{(0)}$. By definition of hidden state alignment, equation 1 is written as $\mathbf{P}_\ell^{-1} x^{(1)}(\ell) = \mathbf{W}_\ell^{(0)} \star \lambda(\mathbf{P}_{\ell-1}^{-1} x^{(0)}(\ell-1))$ where $\mathbf{P}$ is a group element and hence invertible. Since pointwise activation is equivariant under permutation, this is equivalent to $x^{(1)}(\ell) = \mathbf{P}_{\ell-1} \mathbf{W}_\ell^{(0)} \star \left(\mathbf{P}_{\ell-1}^{-1} \lambda(x^{(0)}(\ell-1))\right)$ Finally by equivariance of convolution under channel permutation we obtain $x^{(1)}(\ell) = \left(\mathbf{P}_\ell \mathbf{W}_\ell^{(0)} \mathbf{P}_{\ell-1}^{-1}\right) \star \lambda(x^{(0)}(\ell-1))$. □

Definition 3 (Quotient Space)**.** *The set of equivalent classes is defined as* $[x] = \{y \in M : x \sim y\}$ *where* $x \in M$ *is called a representative. Given a relation* $\sim_P$ *on a space M, the quotient space denoted by* $M/\sim_P$ *or simply* M/P *is defined as the set of equivalent classes* $M/\sim_P := \{[x] : x \in M\} \subset 2^M$.

Conjecture 1 (Fusion by Linear Interpolation)**.** *Two neural network weights* $\mathbf{W}_\ell^{(0)}$ *and* $\mathbf{W}_\ell^{(1)}$ *with the same structure (probably with different channel sizes), which are obtained by different data and initialization* $(\pi^{(0)}, \mu^{(0)})$ *and* $(\pi^{(1)}, \mu^{(1)})$ *respectively, have representatives which can be interpolated so that the loss function remains low on the interpolated weight.*

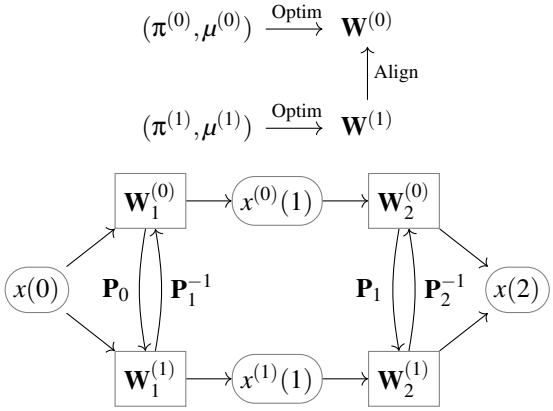

Figure 4: **Model Fusion. Top:** different models (independently trained on different datasets or initializations) coincides in the same loss basin by means of weight alignment. **Bottom:** diagram of the per-layer channel alignment $\mathbf{P}_\ell$.

The alignment is illustrated in figure 4. To facilitate our conjecture of model fusion, a new class of activation functions is designed so that it admits a larger (infinite-order) alignment group namely $(C-1)$-channel rotations around the first axis $P := \{\mathbf{P}_\ell \in \mathbb{R}^{C\times C} : \mathbf{P}_\ell[2,\ldots,C;2,\ldots,C] \in O(C-1)\}$, where $O(C)$ is the set of C-dimensional orthogonal matrices.

3 CONIC LINEAR UNITS

Projective Form Let $x = (x_1, \ldots, x_C)$ be the input of the activation function where C is the network width. Inspired by ReLU

$$\mathrm{ReLU}(x) = \mathrm{Proj}_{V_+}(x) = x_+ \tag{2}$$

whose projected cone $V_+ = \mathbb{R}^C_{\geq 0} = \{x \in \mathbb{R}^C : x_i \geq 0, i = 1,\ldots,C\}$ is the positive cone, CoLU is defined as

$$\mathrm{CoLU}(x) = \mathrm{Proj}_{V\cap H(x)}(x) = \operatorname*{argmin}_{\substack{y\in V \\ (y-x)\perp e}} \|y - x\|_2, \tag{3}$$

where the hypercone is $V = \{x \in \mathbb{R}^C : \|x_e^\perp\|_2 \leq t x \cdot e\}$ with e being the unit vector in the axis direction, $x = x\cdot e + x_e^\perp$, meaning x_e is parallel to e and $x_e^\perp$ is perpendicular to e, and the cone's cross-section hyperplane is $H(x) = \{y \in \mathbb{R}^C : y \cdot e = x \cdot e\}$ whose normal vector is e. $t > 0$ is the tangent value of the opening angle of the cone. We set $t = 1$ and $e = e_1 = (1, 0, \ldots, 0)$ without loss of generality.

In comparison of pointwise activations, CoLU is attending to the round hypercone instead. The two dimensional case is illustrated in figure 5.

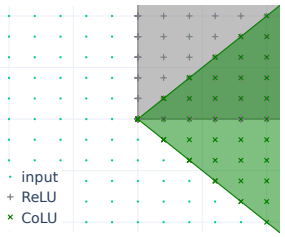

Figure 5: ReLU and CoLU's projective cone in 2D.

Closed Form CoLU is closed-form and auto-differentiable activation as a drop-in replacement of pointwise activations.

$$\begin{aligned}\mathrm{CoLU}(x)_i &= x\mathbb{1}_V + (x_{1+}e_1 + \frac{x_1}{\|x_{-1}\|}x_{-1})\mathbb{1}_{V^c} \\ &= \begin{cases} \mathrm{clamp}(x_1/\|x_{-1}\|, 0, 1)x_i, & i = 2,\ldots,C \\ \max\{x_1, 0\}, & i = 1 \end{cases}\end{aligned} \tag{4}$$

where $x_{-1} = (0, x_2, \ldots, x_C)$, $\mathrm{clamp}(x,a,b) = \min\{\max\{x,a\},b\}$, V^c is the complement set of $V \subseteq \mathbb{R}^C$, and $\|\cdot\|$ is the L^2 norm. In practice, $\|x_{-1}\|$ is replaced with $(\|x_{-1}\| + \varepsilon)$ where ε is a small constant, for numerical stability.

Extensions of Conic Linear Unit

Soft Projection Inspired by Sigmoid-Weighted Linear Units (SiLU) $\mathrm{SiLU}(x) = x\,\mathrm{sigmoid}(x)$, CoLU can be relaxed as a soft projection.

$$\mathrm{CoSiLU}(x)_i = \begin{cases} \mathrm{sigmoid}(x_1/\|x_{-1}\| - 0.5)x_i, & i \geq 2 \\ \max\{x_1, 0\}, & i = 1 \end{cases} \tag{5}$$

L^p-Cones CoLU can also be extended to the case of L^p cones where $p \in \mathbb{R}_+ \cup \{\infty\}$.

$$\mathrm{CoReLU}_p(x)_i = \begin{cases} \mathrm{clamp}(x_1/\|x_{-1}\|_p, 0, 1)x_i, & i \geq 2 \\ \max\{x_1, 0\}, & i = 1 \end{cases} \tag{6}$$

Non-convex Double-Cones Replacing the cone with a signed cone,

$$\mathrm{CoReLU}_{\pm}(x)_i = \begin{cases} \mathrm{clamp}(x_1/\|x_{-1}\|,-1,1)x_i, & i \geq 2 \\ x_1, & i = 1 \end{cases} \tag{7}$$

4 MODEL FUSION

4.1 Model Alignment

With proper permutations, the aligned weights $\mathbf{P}_{\ell-1}^{\top}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell}$ live in the same basin with the reference model relatively. More precisely, define interpolated weight as

$$\mathbf{W}_{\ell}^{(\lambda)} = (1-\lambda)\mathbf{W}_{\ell}^{(0)} + \lambda\mathbf{P}_{\ell}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell-1}^{\top}, \lambda \in [0,1], \tag{8}$$

then the loss barrier $L(\mathbf{W}^{(\lambda)})$ as a function over λ is a concave function for $\mathbf{P}_{\ell} = \mathbf{I}, \forall \ell = 1,\ldots,T$, and this barrier is largely flattened after the alignment, illustrated in figure 8 in the experiment session.

Algorithm 1: Soft Alignment via Optimal Transport

Data: $\mathbf{W}^{(0)}, \mathbf{W}^{(1)}$; // Base and Alternative Models
Result: $\mathbf{P} = \underset{\mathbf{P}_{\ell}\in\Pi(\mathbf{1},\mathbf{1})}{\mathrm{argmax}} \sum_{\ell=1}^{T} \langle \mathbf{W}_{\ell}^{(0)}, \mathbf{P}_{\ell}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell-1}^{\top}\rangle_F$
$\mathbf{P}_{\ell} \leftarrow \mathbf{I}_{C_{\ell}}$; // Initialization
ε ; // Small Constant
$S, S_{\mathrm{prev}} \leftarrow -\infty$;
repeat
  $S_{\mathrm{prev}} \leftarrow S$;
  for $\ell \leftarrow \mathrm{RandPerm}(\{1,\ldots,T-1\})$ **do**
    $\mathbf{P}_{\ell} \leftarrow \underset{\mathbf{P}_{\ell}\mathbf{1}=\mathbf{1}, \mathbf{P}_{\ell}^{\top}\mathbf{1}=\mathbf{1}}{\mathrm{argmax}} \langle \mathbf{W}_{\ell}^{(0)}, \mathbf{P}_{\ell}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell-1}^{\top}\rangle_F + \langle \mathbf{W}_{\ell+1}^{(0)}, \mathbf{P}_{\ell+1}\mathbf{W}_{\ell+1}^{(1)}\mathbf{P}_{\ell}^{\top}\rangle_F$;
  end
  $S \leftarrow \sum_{\ell=1}^{T} \langle \mathbf{W}_{\ell}^{(0)}, \mathbf{P}_{\ell}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell-1}^{\top}\rangle_F$;
until $S \leq S_{\mathrm{prev}} + \varepsilon$;

4.2 Optimal Alignment

The algorithm to find the optimal permutation is maximizing the Frobenius product between the reference model and the aligned model, by using either the weights or activations. Both cases are referred to as

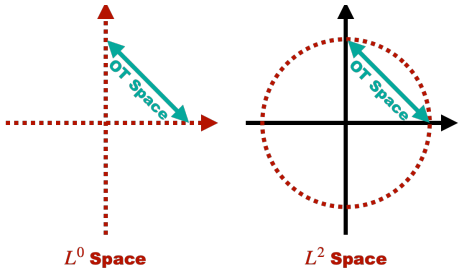

Figure 6: **Geometry** of point-wise activations (L^0) versus CoLU (L^2) and their closeness to OT space.

weight matching and **activation matching** respectively, and we focus on the first method which doesn't require training data. It reduces to a Sum of Bilinear Assignment Problem (SOBAP), and is approximated by solving the Linear Assignment Problem sequentially in a greedy manner, whose convergence is analyzed in (Ainsworth et al., 2023).

Each permutation $\mathbf{P}_{\ell}$ for $\ell = 1,\ldots,T-1$ takes the form of a permutation matrix $P \in \mathcal{S}(C_{\ell})$. The linear assignment problem is stated as

$$\max_{\mathbf{P}_{\ell}\in\mathcal{S}(C_{\ell})} \langle \mathbf{W}_{\ell}^{(0)}, \mathbf{P}_{\ell}\mathbf{W}_{\ell}^{(1)}\mathbf{P}_{\ell-1}^{\top}\rangle_F + \langle \mathbf{W}_{\ell+1}^{(0)}, \mathbf{P}_{\ell+1}\mathbf{W}_{\ell+1}^{(1)}\mathbf{P}_{\ell}^{\top}\rangle_F \tag{9}$$

where the input and output channel spaces are fixed and not aligned, meaning $\mathbf{P}_0 = \mathbf{I}_{C_0}, \mathbf{P}_T = \mathbf{I}_{C_T}$. Equivalently, the linear optimization objective in equation 9 can be written as $\sum_{i,j=1}^{C} \mathbf{C}_{ij}\mathbf{P}_{ij}$.

4.3 Algorithm: Optimal Transport

The constraint of permutation matrices $\mathbf{P}_{\ell} \in \mathrm{S}(C_{\ell})$ can be relaxed to probabilistic assignment matrices whose marginals are ones in both dimensions $\Pi(\mathbf{1},\mathbf{1}) = \{\mathbf{P} \in \mathbb{R}^{C\times C} : \sum_i \mathbf{P}_{ij} = \sum_j \mathbf{P}_{ij} = 1, \forall i,j\}$, also known as bistochastic matrices. This is useful especially in the more general case when the layer width C_{ℓ} differs between the two models, and it can be solved with optimal transport. Using entropic regularization and the Sinkhorn's algorithm, the time complexity can be accelerated from $O(C^2)$ by Linear Programming to $O(C)$ by fixed-point method. The Kantorovich relaxation of optimal transport problem is stated as:

$$\min_{\mathbf{P}\mathbf{1}=\mathbf{1}, \mathbf{P}^{\top}\mathbf{1}=\mathbf{1}} \sum_{i,j=1}^{C} \mathbf{C}_{ij}\mathbf{P}_{ij} + \varepsilon H(\mathbf{P}) \tag{10}$$

where the cost matrix $\mathbf{C}$ is given in equation 9, and $H(\mathbf{P}) = -\mathbf{P}_{ij}(\log \mathbf{P}_{ij} - 1)$ is an entropic regularization term scaled by $\varepsilon > 0$. However, the relaxation of Optimal Transport breaks the model, since a neural network is permutation equivariant, but not channel-wise interpolation equivariant. Figure 6 explains the reason why CoLU leads to seamless fusion, which

is the fact that the distance between interpolation (L^1 level set) and rotation (L^2 level set) spaces are closer than the one between interpolation and permutation. Note that the CoLU symmetry is on $\mathbb{R}^{C-1}$ excluding the first dimension, and $\Pi(\mathbf{1},\mathbf{1})$ is a relaxed constraint and works empirically.

5 EXPERIMENTS

5.1 Generative Performance of CoLU

Performance	SiLU	**CoSiLU**
Symmetry Group	$\mathcal{S}(C)$	$O(C-1)$
# Symmetries	C^T	∞
Time Complexity	$O(C)$	$O(C)$
PSNR / dB (↑)	24.53 ± 2.54	$\mathbf{25.83} \pm 2.52$
IS (↑)	7.23 ± 0.08	$\mathbf{8.04} \pm 0.94$
KID / $\times 10^{-3}$ (↓)	6.35 ± 0.77	$\mathbf{2.71} \pm 0.50$
SSIM / $\times 10^{-1}$ (↑)	8.61 ± 0.55	$\mathbf{8.93} \pm 0.44$

Table 1: Performance of point-wise activation (SiLU) versus CoSiLU (Ours).

Figure 7: Generation results of CoLU-LDM on FFHQ.

Two tasks are experimented: image reconstruction task with autoencoder with regularization of Generative Adversarial Networks (GAN), and in image generation with Diffusion Models (DM).

Image Reconstruction We follow the autoencoder architecture in (Rombach et al., 2022) to build a variational autoencoders enhanced with GAN and perceptual loss regularizations, and replace the Sigmoid Linear Unit (SiLU) activation with CoSiLU in equation 5 The model is trained on the CIFAR10 dataset (Krizhevsky and Hinton,).The models are trained using 8×NVIDIA A100 GPUs until convergence, which typically occurs around $200 \sim 300$ epochs. The hyperparameters involve a base learning rate of 10^{-7}, a batch size of 64, and other parameters as stated in LDM. Multiple evaluation measures are validated including peak signal-to-noise ratio (PSNR), Inception Score (IS) (Salimans et al., 2016), Kernel Inception Distance (KID) (Bińkowski et al., 2018), and Structural Similarity Index Measure (SSIM) (Wang et al., 2004). The results are shown in table 1, where CoSiLU is found to outperform baseline (SiLU) under these criteria.

High-resolusion Image Generation Then we validate CoSiLU on FFHQ (Karras et al., 2019) and AFHQ (Choi et al., 2018) datasets. It is shown that on both datasets, CoSiLU-based diffusion models produces sharp results and work on par with SiLU activation. Generated high-resolution images trained on FFHQ are shown in figure 7.

5.2 Model Fusion

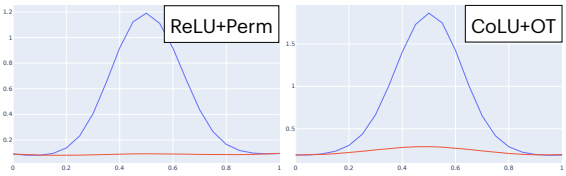

Figure 8: **Seamless Model Fusion.** Loss function along the linear interpolation between two models independently trained from different initializations of fully-connected models on MNIST dataset. The blue line is naïve interpolation, and the red line is interpolation after alignment. **Left:** ReLU network with permutation alignment. **Right:** CoLU network with soft alignment. Both alignments successfully align alternative models towards the base one.

Fusing Recognition Models Alignment of MNIST recognition models are performed on the same network as above with three fully-connected layers and two nonlinearities. As is shown in figure 8, both CoLU and ReLU aligns models with permutation modulus or optimal transport, which applies to networks with varying width.

Fusing Generative Models CoLU was designed aiming at a goal of merging generative models. Figure 9 shows partial milestone towards this goal, namely it's possible to align two super-resolution model so

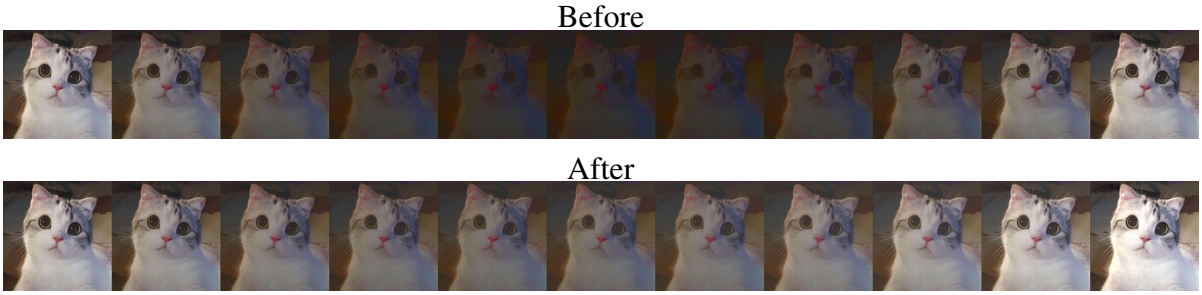

Figure 9: **Fusing generative models.** Outputs of models whose parameters are interpolated between two super-resolution models trained from different initializations. Top row: no alignment. Bottom row: the second model is aligned towards the first model, where intermediate outputs are brighter.

that their interpolation path is closer to the ground-truth of the output. Here it shows a lightening of the output. The super resolution model follows ESPCN (Shi et al., 2016), where all settings follow the original work. We observe an alleviation of the darkening effect after alignment. The base model and the alternative model are initialized with independent and identically distributed weights.

Figure 10 is another result showing that diffusion model might be a more suitable choice than single-pass generative models for fusion via interpolation. The interpolating effect between two dependent diffusion models. Instead of aligning, the right image is output by a model *fine-tuned* from the model on the left, using three images shown on the right. The generated images are of resolution 256x256 without perceptible degradation of quality. The smooth interpolation and sharp outcome presents the feasibility of merging dependent models by linear interpolation.

6 RELATED WORKS

Equivariant Network Convolutional weight sharing (LeCun et al., 1989) is a successful example of imposing spatial homogeneity on the 2D image canvas as a successful a priori assumption upon the network architecture. The introduction of CoLU considers new homogeneity on the *channel* dimension. More generally, (Weiler and Cesa, 2019) conducts a review on equivariant nonlinearities in neural networks and studies certain special forms of nonlinear functions, such as point-wise activations, individual subspace (pixel-wise) activations, norm nonlinearities, etc., which do not cover the case of CoLU.

Disentangling the Channel Dimension Related to the permutation equivariance is the homogeneity of the channel indice's space. It is a crucial subject in convolutional neural networks since it has been proven that local spatial correlation can be sufficiently characterized with deterministic wavelet filters (Bruna and Mallat, 2013). Combined with *learned* pixel-wise linear transform, or 1-by-1 convolution, invariant scattering networks are sufficient to achieve high performance in recognition tasks. Therefore, disentangling the channel dimension is the missing ingredient in simplifying neural networks, where various assumptions can be imposed. For instance, sparsity as strong as block-diagonal in the channel dimension results in group convolution. Orthogonality results in spectrum-preserving weights. In the case of convolutional neural networks, the authors of (Wang et al., 2020b) propose an orthogonality-inducing regularization term to ensure that the convolutional weights are empirically orthogonal. The link to CoLU is close since the symmetric property of CoLU is exactly *pixel-wise* rotation/reflection equivariance, which bridges the gap of the symmetry bottleneck caused by the permutation-restrictive point-wise activations.

Model Alignment Merging trained models by exploiting permutation symmetry of pointwise activations has led to a fruitful line of research (Ashmore and Gashler, 2015; Wang et al., 2020a). Recently in (Ainsworth et al., 2023), deep recognition models like ResNet-50 are also align-able by deterministically matching weights. We continue on this path and further explore the merit of a more symmetric activation function to improve the merging effect, on both recognition and generative models.

7 CONCLUSION

We have introduced a new class of activation functions called Conic Linear Units. Our contribution allows neural networks to possess infinite-order group symmetry beyond channel permutations, which was previously unattainable. This novel design addresses the

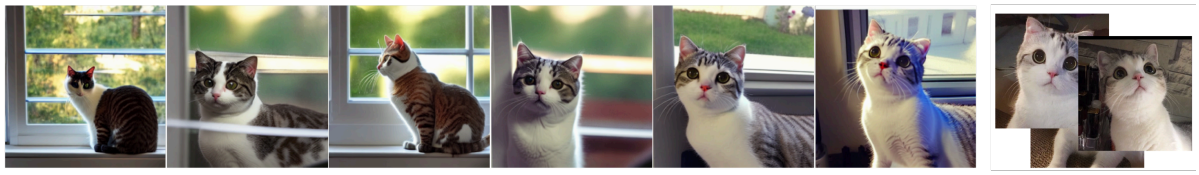

Figure 10: **Fusing fine-tuned models.** Outputs of interpolated models between a Diffusion Model with parameters W_0 and a Diffusion Model with parameters W_1 fine-tuned from the previous model on the dataset of three cat images on the right. More precisely, the 6 left images represent the outputs from a diffusion model whose parameters are $W = (1-\lambda)W_0 + \lambda W_1$ where $\lambda = 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1$. $\lambda = 0$ corresponds to no fine-tuning and $\lambda = 1$ corresponds to the fine-tuned model.

apparent deficiency by incorporating soft-alignment through optimal transport in scenarios like federated learning. Moreover, it outperforms baseline results in terms of image generation quality.

ACKNOWLEDGEMENT

This work was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute). This work was granted access to the HPC resources of IDRIS under the allocation 2022-AD011013178 made by GENCI, and supported by cloud TPU from Google's TPU Research Cloud (TRC). We thank Irène Waldspurger, Gabriel Peyré and Grégoire Szymanski for insightful suggestions on the draft of the paper.

REFERENCES

Ainsworth, S., Hayase, J., and Srinivasa, S. (2023). Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representations*.

Ashmore, S. and Gashler, M. (2015). A method for finding similarity between multi-layer perceptrons by forward bipartite alignment. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.

Bińkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A. (2018). Demystifying MMD GANs. In *International Conference on Learning Representations*.

Bruna, J. and Mallat, S. (2013). Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886.

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797.

Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410.

Krizhevsky, A. and Hinton, G. Technical report, University of Toronto, Toronto, Ontario.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. (1989). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems*, 2.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., and Chen, X. (2016). Improved techniques for training gans. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, Los Alamitos, CA, USA. IEEE Computer Society.

Singh, S. P. and Jaggi, M. (2020). Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33:22045–22055.

Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D., and Khazaeni, Y. (2020a). Federated learning with matched averaging. In *International Conference on Learning Representations*.

Wang, J., Chen, Y., Chakraborty, R., and Yu, S. X. (2020b). Orthogonal convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11505–11515.

Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

Weiler, M. and Cesa, G. (2019). General e (2)-equivariant steerable cnns. *Advances in Neural Information Processing Systems*, 32.