Self-supervised Transformation Learning for Equivariant Representations

Jaemyung Yu[1] **Jaehyun Choi**[1] **Dong-Jae Lee**[1] **HyeongGwon Hong**[1] **Junmo Kim**[1]
[1]Korea Advanced Institute of Science and Technology (KAIST)
{jaemyung,chlwogus,jhtwosun,honggudrnjs,junmo.kim}@kaist.ac.kr

Abstract

Unsupervised representation learning has significantly advanced various machine learning tasks. In the computer vision domain, state-of-the-art approaches utilize transformations like random crop and color jitter to achieve invariant representations, embedding semantically the same inputs despite transformations. However, this can degrade performance in tasks requiring precise features, such as localization or flower classification. To address this, recent research incorporates equivariant representation learning, which captures transformation-sensitive information. However, current methods depend on transformation labels and thus struggle with interdependency and complex transformations. We propose Self-supervised Transformation Learning (STL), replacing transformation labels with transformation representations derived from image pairs. The proposed method ensures transformation representation is image-invariant and learns corresponding equivariant transformations, enhancing performance without increased batch complexity. We demonstrate the approach's effectiveness across diverse classification and detection tasks, outperforming existing methods in 7 out of 11 benchmarks and excelling in detection. By integrating complex transformations like AugMix, unusable by prior equivariant methods, this approach enhances performance across tasks, underscoring its adaptability and resilience. Additionally, its compatibility with various base models highlights its flexibility and broad applicability. The code is available at `https://github.com/jaemyung-u/stl`.

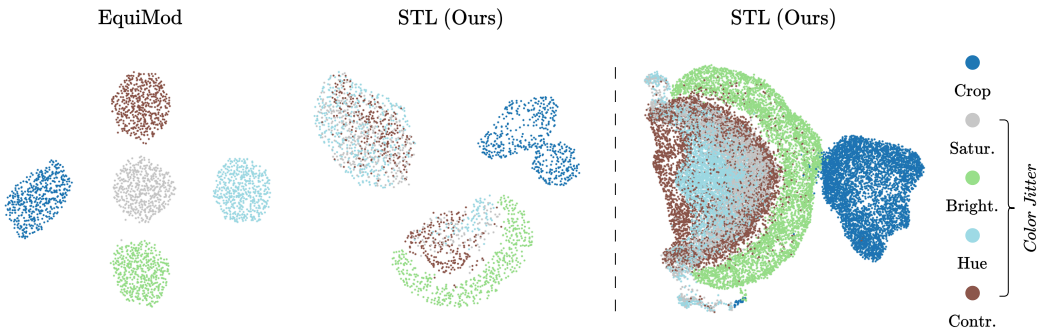
Keyword

Equivariant Learning, Transformation Representation, Self-supervised Transformation Learning

1 Introduction

Recently, unsupervised representation learning [15, 10, 34] has made remarkable strides in various machine learning tasks and is actively employed as the ground model. In particular to the state-of-the-art computer vision models [4, 18, 3, 5, 48, 1], invariant representation learning utilizes various augmentations, hereinafter *transformations*, including but not limited to random crop, horizontal flip, color jitter, and Gaussian blur. Their objective is to embed the semantically same inputs obtained through transformation, based on the notion that semantic differences due to the various transformations are inconsequential. Unfortunately, although effective most of the time, it does not always guarantee performance gain in the target downstream tasks. For instance, in tasks such as localization or flower classification, applying random crop or color jitter for invariant learning degrades performance by diluting discriminative features regarding the position of the object or the color of the flower, respectively.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) Equivariant Transformation (b) Transformation Representation

Figure 1: **Visualization of Equivariant Transformation and Transformation Representation.** (Left) UMAP [32] visualizations of functional weights from equivariant transformations implemented with a hypernetwork. EquiMod uses transformation labels to generate these weights, while STL derives them from the representation pairs of transformed and original image. (Right) UMAP visualizations of transformation representations obtained from representation pairs of original input image and transformed input image.

To circumvent such a problem while harnessing the benefits of invariant representation learning, the research has transitioned towards incorporating equivariant representation learning [8, 28, 36, 9, 13] alongside invariant representation learning. As learning an equivariant representation in parallel with an invariant representation requires the representation to be responsive to transformations, the objective of equivariant representation learning is to capture transformation-sensitive information. This can be achieved either by training the model to predict transformations through representation pairs as in E-SSL [8] and AugSelf [28], referred to as *implicit equivariant learning*, or by learning the corresponding equivariant transformations in the representation space as in SEN [36], EquiMod [9], and SIE [13], referred to as *explicit equivariant learning*. However, these methods face a major issue due to the requirement of transformation labels. Firstly, optimizing with a transformation label, each transformation is treated independently, disregarding interdependency among transformations. Consequently, each component in color jitter transformation is treated distinctively although they are related to each other in the sense that they are applying transformation in the color as shown in Figure 1 (a). Secondly, due to the limited expressiveness of transformation labels, it fails to encompass complex transformations such as AugMix [23], where the sequences of transformations are randomly applied with varying weights. After all, the reliance on the transformation label limits the performance gain in equivariant representation learning.

To address the aforementioned limitations, we propose Self-supervised Transformation Learning (STL) for learning representation of transformation, which enhances the potential of equivariant learning. In STL, the transformation label is replaced by the transformation representation derived from the representation pairs of original and transformed image. To ensure the transformation representation captures the information of transformation, we train it to be invariant to the same transformation applied to various images, similar to how contrastive learning trains image representation to be invariant to various transformations. Based on the derived transformation representations, we aim to learn the corresponding equivariant transformations in the representation space. To avoid the *trivial solution* when learning the equivariant transformation, we apply the transformation representation obtained from another image but with the same transformation. Through optimization, STL is able to learn the transformation representation with the same batch complexity as previous methods.

We demonstrate the effectiveness of STL in transfer learning across various datasets and tasks, including both classification and detection. Additionally, by incorporating AugMix, a complex transformation that is not feasible with existing equivariant learning, STL enhances performance across all tasks, showcasing its broad applicability. STL's compatibility with diverse base models, further highlights its versatility, as it achieves the highest average accuracy across foundational models. Extensive experiments and ablation studies further validate STL's ability to capture interdependencies among transformations in an unsupervised manner. This is evident not only in transformation representations and equivariant transformation clustering, which reveal nuanced relationships between transformations (see Figure 1), but also in its superior performance in transformation prediction, surpassing existing equivariant learning methods.

2 Preliminaries: Transformation Invariant and Equivariant Learning

Transformation as Group Action. A group G consists of elements and an operation that satisfies closure, associativity, the existence of an identity element e, and the existence of inverses for all elements. The group action of G on a set X is defined as a function $\cdot : G \times X \to X$, which ensures the identity operation $e \cdot x = x$ for all $x \in X$ and maintaining the compatibility of the operations $(g \cdot h) \cdot x = g \cdot (h \cdot x)$ for all $g, h \in G$. In the context of image processing, transformations can be viewed as a group action where the group T consists of transformations, and the set X is the collection of images. For example, if $R \in T$ is a rotation and $F \in T$ is a flip transformation, applying R followed by F to an image $x \in X$ is represented as $F \cdot (R \cdot x) = F(R(x))$. This structure ensures that image transformations are consistent and reversible.

Transformation Invariant Representation. In the context of transformation as group action, the *transformation invariance* of an image representation obtained through an encoder $f : X \to Y$, which maps the set of images X to representation space Y, is defined as follows: Let T be a group of transformations applied to an input image $x \in X$, with $t(x)$ representing the image after transformation $t \in T$. The representation $f(x)$ is invariant to all transformations in T if $f(x)$ of an input image x remains unchanged even after applying any transformation t to x.

$$f(x) = f(t(x)) \quad \forall t \in T. \tag{1}$$

In self-supervised learning, leveraging transformation invariance is crucial for learning representations, as it aligns transformed input images within the representation space. The objective of transformation invariant representation learning $\mathcal{L}_{\text{inv}}$ is formalized with a dissimilarity loss $\mathcal{L}$ as follows:

$$\min_f \mathbb{E}_{x,t}\Big[\mathcal{L}_{\text{inv}}(x,t)\Big] \quad \text{s.t.} \quad \mathcal{L}_{\text{inv}}(x,t) = \mathcal{L}(f(x), f(t(x))), \tag{2}$$

where $\mathcal{L}$ represents the dissimilarity metric between representations. In contrastive learning, $\mathcal{L}$ is instantiated as metrics like the InfoNCE [35] loss, which reduces the distance between representations of semantically similar inputs while increasing the gap between those of dissimilar inputs.

Transformation Equivariant Representation. Extending the concept of invariance, *transformation equivariance* ensures that an image representation changes predictably according to the applied transformation. The representation $f(x)$ is equivariant to all transformations in T if there exists a group action $\phi : T \times Y \to Y$ on the representation space Y, ensuring that the application of t to an image x leads to a corresponding transformation $\phi(t, f(x))$ in its representation.

$$f(t(x)) = \phi(t, f(x)) \quad \forall t \in T. \tag{3}$$

The function $\phi(t, \cdot) : Y \to Y$ is referred to as the equivariant transformation on the representation space corresponding to the transformation t. When the equivariant transformation $\phi(t, \cdot)$ becomes the identity, it implies transformation invariance, showcasing the model's insensitivity to transformations. In EquiMod [9] and SIE [13], equivariant transformation $\phi(\hat{t}, \cdot)$ is represented by a network parameterized by the corresponding transformation label $\hat{t}$. This network is trained to associate each transformation t with its corresponding equivariant transformation $\phi(\hat{t}, \cdot)$ in the representation space. The objective of transformation equivariant representation learning $\mathcal{L}_{\text{equi}}$ is formalized as follows:

$$\min_{f,\phi} \mathbb{E}_{x,t}\Big[\mathcal{L}_{\text{equi}}(x,t)\Big] \quad \text{s.t.} \quad \mathcal{L}_{\text{equi}}(x,t) = \mathcal{L}(\phi(\hat{t}, f(x)),\ f(t(x))). \tag{4}$$

3 Equivariant Learning with Self-supervised Transformation Learning

3.1 Equivariant Learning without Transformation Label

Transformation Representation. Instead of relying on transformation labels that require knowledge of the transformation group structure, we propose leveraging pairs of representations derived from original images and their transformed counterparts to implicitly represent transformations. We introduce an auxiliary encoder $f_T : Y \times Y \to Y_T$ designed to process pairs of representations $(f(x), f(t(x)))$, where $f(x)$ and $f(t(x))$ are the representation of the original image and of the transformed image respectively. This encoder outputs a transformation representation $y_t^x \in Y_T$, capturing the inherent transformation between the original and transformed images without explicit transformation labels.

$$y_t^x = f_T(f(x),\ f(t(x))) \in Y_T \quad \text{for } t \in T \text{ and } x \in X. \tag{5}$$

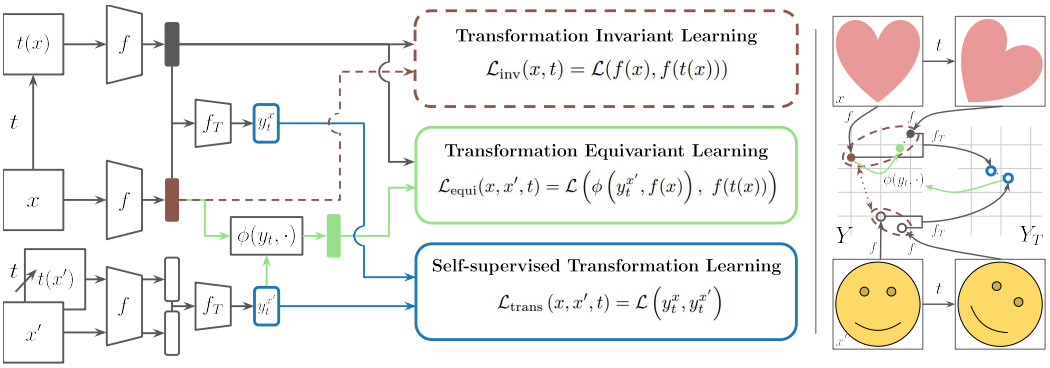


Figure 2: **Transformation Equivariant Learning with Self-supervised Transformation Learning.** (Left) The overall framework of STL. For given image and transformations, it demonstrates: 1) transformation invariant learning, which aligns the representations of image and transformed image; 2) transformation equivariant learning, where the representation of image transformed by an equivariant transformation (obtained from the transformation representation of different image with the same applied transformation) aligns with the transformed image's representation; 3) self-supervised transformation learning, which aligns the transformation representations obtained from different image pairs. (Right) It illustrates the transformations of each representation and the equivariant transformations within the representation space.

Equivariant Learning with Transformation Representation. In contrast to methods directly utilizing transformation labels to learn equivariant transformations in the representation space, our proposed approach substitutes labels with transformation representations derived from pairs of images. However, there is a risk of encountering a trivial solution when using the transformation representation derived from the representation pair of the same image for which the equivariant transformation is applied. Specifically, the equivariant transformation might simply output the same representation $f(t(x)) = \phi(f_T(f(x), f(t(x))), f(x))$ that was used to obtain the transformation representation. To address this issue, we propose using transformation representations $y_t^{x'}$ derived from pairs of a different image x' to apply equivariant transformations. Therefore, the transformed representation of the image through equivariant transformation in the representation space can be expressed as follows:

$$\phi\left(y_t^{x'},\ f(x)\right) = \phi\left(f_T\left(f\left(x'\right), f\left(t(x')\right)\right),\ f(x)\right) \quad \text{for } x \neq x' \in X. \tag{6}$$

Leveraging this approach, it is possible to learn transformation equivariant representations without explicit transformation labels, through the following objective:

$$\min_{f, f_T, \phi} \mathbb{E}_{x \neq x', t}\left[\mathcal{L}_{\text{equi}}(x, x', t)\right] \quad \text{s.t.} \quad \mathcal{L}_{\text{equi}}(x,x',t) = \mathcal{L}\left(\phi\left(y_t^{x'}, f(x)\right),\ f(t(x))\right). \tag{7}$$

3.2 Self-supervised Transformation Learning (STL)

We hypothesize that transformation representations, y_t^x, derived from an input image x and its transformed image $t(x)$, encode the transformation t independently of the input image x. Nonetheless, ensuring image-invariant encoding of t is not trivial.

$$y_t^x = y_t^{x'} \quad \forall x \neq x' \in X. \tag{8}$$

To address this, we introduce Self-supervised Transformation Learning (STL), which adapts contrastive learning for transformation representation. Contrary to contrastive learning, which aims to align representations $f(x)$ of the same image under different transformations to promote transformation invariance, STL instead focuses on aligning transformation representations, y_t^x and $y_t^{x'}$, derived from different images $x \neq x'$ subjected to the identical transformation t. The objective of STL is formalized as follows:

$$\min_{f, f_T} \mathbb{E}_{x \neq x', t}\left[\mathcal{L}_{\text{trans}}\left(x, x', t\right)\right] \quad \text{s.t.} \quad \mathcal{L}_{\text{trans}}\left(x,x',t\right) = \mathcal{L}\left(y_t^x, y_t^{x'}\right). \tag{9}$$

Hereinafter, equivariant learning using transformation representations learned through self-supervised transformation learning will simply be referred to as STL.

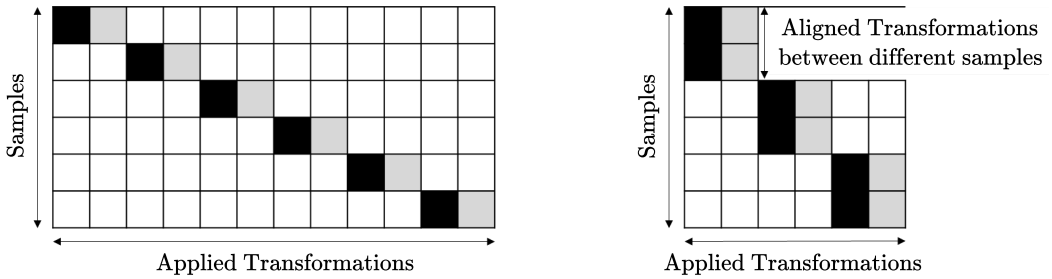


Figure 3: **Aligned Transformed Batch.** (Left) In self-supervised learning methods, batch compositions typically involve applying two different transformations to each input image. (Right) In STL, batches are composed by pairing two images together, and applying the same transformation pair.

3.3 Implementation Details

Dissimilarity Metric. We use the InfoNCE loss from SimCLR [4] for the formulation and implementation of STL. Our methodology is not limited to this model, as demonstrated by ablation study, which shows the feasibility of applying our approach across various self-supervised learning models, such as BYOL [18], SimSiam [5], and Barlow Twins [48], through straightforward extensions (see Appendix A). Like EquiMod [9], we employ specialized projectors g_{inv}, g_{equi}, and g_{trans} to map representations into distinct embedding spaces Z_{inv}, Z_{equi}, and Z_{trans}, aligned with the objectives of invariant, equivariant, and transformation representation learning. We adopt the InfoNCE loss as the dissimilarity metric across these spaces, similar to the approach in SimCLR. The InfoNCE loss function is defined as follows:

$$\mathcal{L}_{\text{InfoNCE}}\left(y, y^{+}; g, \tau, \{y\}_i\right) = -\log \frac{\exp\left(\text{sim}\left(g(y), g(y^{+})\right)/\tau\right)}{\sum_{y_i \neq y} \exp\left(\text{sim}\left(g(y), g(y_i)\right)/\tau\right)}, \tag{10}$$

where y represents representation of an input image, y^{+} denotes the corresponding representation to align, $\text{sim}(\cdot)$ indicates a similarity function, and τ is a temperature scaling parameter. For simplicity, batch $\{y\}_i$ are omitted in the subsequent loss functions. We define three specific loss functions for invariant, equivariant, and transformation representation learning, each building on the InfoNCE loss:

$$\mathcal{L}_{\text{inv}}(x, t) = \mathcal{L}_{\text{InfoNCE}}\big(f(x), f(t(x));\ g_{\text{inv}}, \tau_{\text{inv}}\big), \tag{11}$$

$$\mathcal{L}_{\text{equi}}(x, x', t) = \mathcal{L}_{\text{InfoNCE}}\big(\phi\big(y_t^{x'}, f(x)\big),\ f(t(x));\ g_{\text{equi}}, \tau_{\text{equi}}\big), \tag{12}$$

$$\mathcal{L}_{\text{trans}}(x, x', t) = \mathcal{L}_{\text{InfoNCE}}\big(y_t^{x},\ y_t^{x'};\ g_{\text{trans}}, \tau_{\text{trans}}\big). \tag{13}$$

Aligned Transformed Batch. To implement STL, we need transformation representations obtained from different images. Unlike typical batch configurations in contrastive learning, where different transformations are applied to each input image, we construct batches by applying identical transformations to image pairs, as illustrated in Figure 3. In our approach, each image undergoes two distinct transformations, denoted t and t', but for simplicity, we consider only a single direction of transformation, treating t as equivalent to $t' \cdot t^{-1}$. This aligned transformed batch configuration maintains the same computational complexity as typical contrastive learning setups while preserving input diversity. It also increases the count of identical transformations applied across different images, which is essential for transformation learning, without diminishing input diversity. To assess computational costs, we measured forward-backward time over 1000 iterations following a 1000-iteration warm-up. With an auxiliary network and loss calculation, our approach required only about 10% more time per iteration than SimCLR, which focuses solely on invariant learning, as shown in Table 1.

Table 1: **Computational Cost.** Forward-backward time per iteration on NVIDIA 3090 GPU with ResNet-50 and batch size 256.

Method	Time (s)	Ratio
SimCLR	0.51	1.00
AugSelf	0.51	1.00
EquiMod	0.53	1.01
STL (Ours)	0.56	1.11

Overall Objective. The overall framework of STL is shown in Figure 2. Using aligned transformed batch, along with the InfoNCE loss, we define the overall objective with the hyperparameters λ_{inv}, λ_{equi} and λ_{trans} for balancing the respective losses as follows:

$$\min_{f, f_T, \phi} \mathbb{E}_{x \neq x', t}\Big[\lambda_{\text{inv}}\mathcal{L}_{\text{inv}}(x, t) + \lambda_{\text{equi}}\mathcal{L}_{\text{equi}}(x, x', t) + \lambda_{\text{trans}}\mathcal{L}_{\text{trans}}(x, x', t)\Big]. \tag{14}$$

Table 2: **Out-domain Classification.** Evaluation of representation generalizability on the out-domain downstream classification tasks. Linear evaluation accuracy (%) is reported for ResNet-50 pretrained on ImageNet100.

Method	CIFAR10	CIFAR100	Food	MIT67	Pets	Flowers	Caltech101	Cars	Aircraft	DTD	SUN397	Mean
Invariant Learning :												
SimCLR	84.24	64.15	59.00	54.78	58.95	91.58	79.32	27.07	36.00	66.01	42.77	60.35
with AugMix	86.90	**67.70**	62.90	57.24	63.75	93.16	83.67	32.37	43.17	67.93	46.15	64.09
Implicit Equivariant Learning :												
E-SSL	85.09	65.74	60.91	56.64	61.00	92.31	80.77	28.84	38.04	66.38	43.49	61.75
AugSelf	85.55	66.09	62.63	57.16	62.61	93.41	82.33	30.71	40.35	68.51	45.24	63.14
Explicit Equivariant Learning :												
SEN	80.68	56.53	52.50	46.79	45.27	79.24	73.42	14.41	27.51	57.45	33.51	51.57
EquiMod	82.89	61.36	56.38	52.84	52.68	87.42	79.17	22.02	34.62	64.10	39.86	57.58
SIE	81.72	58.49	54.04	49.70	47.21	84.37	74.39	16.71	31.68	59.20	35.29	53.89
STL (Ours)	86.55	66.84	64.32	56.64	65.00	94.51	81.83	35.44	45.42	64.68	44.69	64.18
with AugMix (Ours)	**87.19**	**67.70**	**66.12**	**59.70**	**67.10**	**94.87**	**84.61**	**38.48**	**46.14**	**69.57**	**45.75**	**66.11**

4 Experiments

Baselines. We compare STL with implicit and explicit equivariant learning methods, using SimCLR [4] as the base invariant model. Implicit methods (E-SSL [8] and AugSelf [28]) learn equivariant representations via transformation prediction tasks. Explicit methods (SEN [36], EquiMod [9], and SIE [13]) use transformation labels for equivariant learning. All methods, including STL, are trained and evaluated with SimCLR as the base model. Experiments with other base models are included in the ablation study.

Datasets. We pretrain on STL10 [7] with ResNet-18 and ImageNet100 [39, 42] with ResNet-50, following the split in [42]. Evaluation spans 11 downstream classification tasks (CIFAR10/100 [27], Food [2], MIT67 [38], Pets [37], Flowers [33], Caltech101 [12], Cars [26], Aircraft [31], DTD [6], SUN397 [47]) with a linear transfer protocol [41]. For detection, VOC07+12 dataset [11] and the protocol from [48] are used. Dataset and protocol details are in Appendix B and C.

Table 3: **In-domain Classification.** Evaluation of representation on in-domain classification task. Linear evaluation accuracy (%) is reported for ResNet-50 pretrained on ImageNet100.

Method	In-domain
Invariant Learning :	
SimCLR	81.20
SimCLR with AugMix	80.54
Implicit Equivariant Learning :	
E-SSL	**82.10**
AugSelf	81.08
Explicit Equivariant Learning :	
SEN	76.32
EquiMod	80.70
SIE	79.40
STL (Ours)	81.10
STL with AugMix (Ours)	81.64

Setup. For STL and explicit baselines (SEN, EquiMod, and SIE), we use an equivariant transformation network with a hypernetwork based on SIE. In SEN, EquiMod, and SIE, the hypernetwork uses transformation labels; in STL, it leverages transformation representations. STL also includes a 3-layer MLP with a 512-dimensional hidden layer to encode 128-dimensional transformation representations from input pairs. Equivariant transformations include random crop and color jitter, with other transformations applied randomly, consistent with typical contrastive learning. The transformation prediction loss weight is set to 0.5 for implicit baselines, and the equivariant learning weight is set to 1 for explicit baselines. STL uses weights of 1, 1, and 0.2 for invariant, equivariant, and transformation learning losses, respectively. We apply AugMix [23] to evaluate STL's adaptability to complex transformations, incompatible with other methods. Details on transformation labels in standard equivariant learning are in Appendix D. All analyses and ablations, except main experiments, use STL10-pretrained models. Additional setup details are in Appendix E.

4.1 Main Results

Image Classification. To assess generalizability, we apply the linear evaluation protocol on various downstream tasks. As shown in Table 2, STL outperforms existing methods on 7 out of 11 datasets. With AugMix, a complex transformation combination, STL achieves the highest performance across all datasets, underscoring its ability to generalize across diverse transformations, even those without explicit labels, to improve generalization. In Table 3, STL shows a minimal trade-off on in-domain tasks compared to SimCLR, with only a slight decrease from 81.20% to 81.10%, a smaller trade-off than other explicit equivariant models. Combined with AugMix, STL reaches 81.64%, showing adaptability to complex transformations and further enhancing in-domain performance. Results on STL10-pretrained models are provided in the Appendix F.

Table 4: **Object Detection.** Evaluation of representation generalizability on a downstream object detection task. Average precision is reported for ImageNet100-pretrained ResNet-50 fine-tuned on VOC07+12.

Method	AP_{all}	AP_{50}	AP_{75}
SimCLR	45.67	72.50	47.83
AugSelf	45.99	72.46	49.23
EquiMod	51.55	78.03	56.17
STL (Ours)	51.95	78.34	56.96
with AugMix (Ours)	**52.70**	**78.81**	**57.76**

Table 5: **Transformation Prediction.** Evaluation of transformation representation from learned represetation pairs. Regression tasks use MSE loss, and transformation type classification uses accuracy.

	Regression (↓)			Classification (↑)
Method	Crop	Color	All	Trans. Type
SimCLR	0.02	0.13	0.08	68.54
AugSelf	**0.01**	0.04	0.03	88.49
EquiMod	**0.01**	0.07	0.04	82.20
STL (Ours)	**0.01**	**0.03**	**0.02**	**93.67**

Object Detection. We evaluate STL on the VOC07+12 object detection task. As shown in Table 4, STL outperforms the invariant learning model SimCLR, as well as AugSelf, a representative model for implicit equivariant learning, and EquiMod, a representative for explicit equivariant learning, across all metrics: AP_{all}, AP_{50} and AP_{75}. STL achieves 51.95 in AP_{all}, and with AugMix, further improves to 52.70 in AP_{all}, 78.81 in AP_{50}, and 57.76 in AP_{75}, demonstrating robust adaptability to complex transformations and enhanced precision in localization.

4.2 Analysis

Transformation Representation. To evaluate learned transformation representations of STL, we assess both parameter prediction and type classification on test images with transformations used during pretraining, specifically crop and color jitter. Parameter prediction uses MSE loss to measure accuracy in predicting transformation specifics, while type classification assesses the model's ability to distinguish among transformation categories. As shown in Table 5, STL achieves the lowest MSE for crop, color jitter, and the combined metric, indicating precise capture of transformation details. In classification, STL attains 93.67% accuracy, surpassing other models, even though AugSelf incorporates parameter regression into its learning. This demonstrates STL's robust generalization in transformation learning without direct parameter supervision.

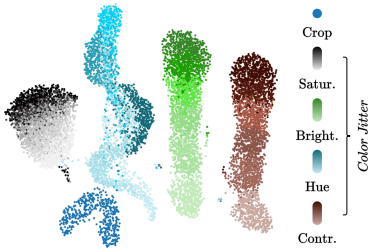


Figure 4: **Visualization of Transformation Representations by Intensity.** UMAP visualization of transformation representations organized by intensity levels for each transformation type, including random crop and color jitter variations in brightness, contrast, saturation, and hue. Parameter ranges for each transformation are divided into four segments to apply varying intensities. Representations are captured by a ResNet-18 model pretrained on STL10 with a transformation backbone.

Additionally, a qualitative UMAP visualization of transformation representations from test image pairs reveals STL's understanding of transformation relationships. Figure 1(b) shows distinct clusters for each transformation type, including crop, brightness, contrast, saturation, and hue, with color-related transformations grouped closely, reflecting STL's capture of inter-relationships among similar transformations. Figure 4 shows that transformations with similar intensity levels are positioned closer in the representation space, forming continuous representations that capture the intrinsic order of intensity within each transformation type. This structure indicates STL's effective learning of transformation intra-relationships, demonstrating its nuanced understanding of both type and intensity.

Transformation Equivariance. We evaluate the accuracy of STL's learned equivariant transformations in reflecting real transformations in image space. This evaluation involves applying 60 transformations per image from the STL10 test dataset, including standalone transformations like crop and color jitter, as well as the standard combinations used during training. For each transformed representation, we rank other representations by similarity, ordering them from closest to furthest. Metrics include Mean Reciprocal Rank (MRR), Hit@k (H@k), and Precision (PRE). MRR is defined as the mean of reciprocal ranks $1/r$, where r is the rank of the nearest representation corresponding to the designated transformation. Hit@k calculates the probability $P(r \leq k)$ that the correct transformation rank r is within the top k. PRE is the mean squared error between the representation from the top-ranked transformation and the actual transformation's parameter vector.

Table 6: **Transformation Equivariance.** Evaluation of the equivariant transformation. Mean Reciprocal Rank (MRR), Hit@k (H@k), and Precision (PRE) metrics on various transformations (crop and color jitter).

	Crop				Color				All			
Method	MRR(↑)	H@1(↑)	H@5(↑)	PRE(↓)	MRR(↑)	H@1(↑)	H@5(↑)	PRE(↓)	MRR(↑)	H@1(↑)	H@5(↑)	PRE(↓)
SEN	0.34	0.15	0.58	0.14	0.18	0.05	0.31	3.69	0.22	0.08	0.37	2.70
EquiMod	**0.37**	0.17	**0.60**	**0.13**	0.16	0.05	0.28	3.72	0.22	0.09	0.36	2.72
SIE	0.33	0.14	0.55	0.33	0.17	0.05	0.28	3.70	0.21	0.08	0.35	2.74
w/o $\mathcal{L}_{\text{trans}}$ (Ours)	0.31	0.18	0.46	0.69	0.27	0.13	0.40	3.37	0.29	0.16	0.43	2.50
STL (Ours)	**0.37**	**0.22**	0.54	0.64	**0.33**	**0.18**	**0.52**	**2.76**	**0.36**	**0.21**	**0.53**	**2.07**

Table 7: **Loss Function Ablation Study.** Image classification and transformation prediction results of ResNet-18 pretrained on STL10 with selective inclusion of loss terms for invariant learning ($\mathcal{L}_{\text{inv}}$), equivariant learning ($\mathcal{L}_{\text{equi}}$), and self-supervised transformation learning ($\mathcal{L}_{\text{trans}}$). For image classification, in-domain accuracy (%) and the average accuracy (%) across multiple out-domain datasets are shown. For transformation prediction, MSE is used for regression of crop and color transformations, and accuracy (%) is used for transformation type classification.

	Loss Functions			Image Classification		Transformation Prediction	
Method	$\mathcal{L}_{\text{inv}}$	$\mathcal{L}_{\text{equi}}$	$\mathcal{L}_{\text{trans}}$	In-domain (↑)	Out-domain (↑)	Regression (↓)	Classification (↑)
Only Invariance	✓	-	-	84.74	43.11	0.08	68.54
Only Equivariance	-	✓	-	83.53	**49.99**	**0.02**	93.54
STL w/o $\mathcal{L}_{\text{inv}}$	-	✓	✓	81.86	48.62	**0.02**	93.54
STL w/o $\mathcal{L}_{\text{equi}}$	✓	-	✓	80.99	47.30	**0.02**	**93.92**
STL w/o $\mathcal{L}_{\text{trans}}$	✓	✓	-	**85.11**	48.49	0.08	69.57
STL	✓	✓	✓	84.83	**49.97**	**0.02**	93.67

As shown in Table 6, STL outperforms previous methods in most metrics, indicating closer alignment of its equivariant transformations with real image transformations in representation space. Notable exceptions are crop H@5 and PRE, where other methods perform slightly better. Overall, STL captures both individual and combined transformations more faithfully, achieving robust alignment between representation and image transformations. Self-supervised transformation learning, by maintaining image-invariance, enhances input consistency for equivariant transformation learning and significantly improves alignment accuracy. Without this component, STL's ability to capture transformation nuances declines, underscoring the role of each component in the STL framework.

4.3 Ablation Studies

Loss Functions. We conduct an ablation study to analyze the impact of each loss function on STL's performance across image classification and transformation prediction tasks. Specifically, we examine the contributions of invariant learning ($\mathcal{L}_{\text{inv}}$), equivariant learning ($\mathcal{L}_{\text{equi}}$), and self-supervised transformation learning ($\mathcal{L}_{\text{trans}}$) by selectively removing each loss term. Table 7 illustrates a clear trade-off between invariance and equivariance in STL's performance. The Only Invariance configuration achieves high in-domain accuracy of 84.74% but suffers from low out-domain accuracy of 43.11% and limited transformation prediction capabilities, highlighting restricted generalizability. In contrast, Only Equivariance improves out-domain accuracy to 49.99% and achieves strong transformation prediction with an MSE of 0.02, indicating enhanced generalization and transformation awareness, albeit with a slight reduction in in-domain performance.

When $\mathcal{L}_{\text{trans}}$ is omitted, the model maintains high in-domain accuracy but exhibits weak out-domain and transformation performance, suggesting that the absence of transformation representation learning leads to a focus on invariance. Excluding $\mathcal{L}_{\text{inv}}$ improves out-domain accuracy to 47.30% and enhances transformation alignment by preventing collapse into pure invariance, although this comes at a moderate cost to in-domain accuracy. Without $\mathcal{L}_{\text{equi}}$, the in-domain performance decreases further as transformation learning alone lacks sufficient structure for alignment. Finally, the full STL, which incorporates all three losses, achieves the best balance, with superior performance across in-domain and out-domain tasks and optimal transformation prediction results. This configuration minimizes in-domain trade-offs while capturing a comprehensive view of transformations, ensuring alignment between representation and image transformations across both in-domain and out-domain tasks.

Table 8: **Transformation Ablation Study.** Linear evaluation accuracy (%) of ResNet-18 pretrained on STL10 with various transformations used as equivariance targets.

Trans.	Method	CIFAR10	CIFAR100	Food	MIT67	Pets	Flowers	Caltech101	Cars	Aircraft	DTD	SUN397	Mean
crop	AugSelf	82.89	54.92	33.19	**39.70**	44.40	64.96	67.63	15.58	25.38	**41.86**	27.89	45.31
	EquiMod	83.76	55.33	32.01	37.76	41.65	63.00	66.28	14.18	24.96	41.54	26.46	44.27
	STL	**84.94**	**59.12**	**35.15**	39.40	**45.35**	**68.38**	**70.78**	**17.96**	**33.00**	**41.86**	**28.71**	**47.70**
color	AugSelf	**84.33**	57.47	36.57	39.40	**46.80**	71.18	67.91	17.03	**27.12**	43.83	29.37	47.36
	EquiMod	82.22	51.77	31.21	34.18	39.57	61.17	62.07	12.51	21.36	39.52	23.48	41.73
	STL	84.16	**58.71**	**38.49**	**41.34**	45.90	**74.36**	**68.48**	**17.31**	**27.12**	**46.54**	**31.17**	**48.51**
crop + color	AugSelf	84.26	57.78	36.82	40.30	45.46	73.38	68.11	17.22	27.63	45.96	30.38	47.94
	EquiMod	81.35	51.86	33.91	37.76	41.92	66.18	67.38	15.22	25.80	42.50	26.70	44.60
	STL	**85.37**	**61.05**	**39.41**	**41.27**	**46.58**	**76.43**	**71.47**	**19.04**	**30.75**	**46.17**	**32.13**	**49.97**
all	AugSelf	81.76	54.90	36.51	40.90	46.17	71.43	70.14	**18.63**	**30.96**	**45.21**	30.40	47.91
	EquiMod	84.42	56.65	34.23	37.99	42.98	67.16	68.41	15.18	26.91	43.94	26.97	45.89
	STL	**84.96**	**58.91**	**36.71**	**42.09**	**46.25**	**72.41**	**71.01**	17.72	28.44	43.83	**30.99**	**48.48**

Table 9: **Base Invariant Learning Model Ablation Study.** Linear evaluation accuracy (%) of ResNet-18 pretrained on STL10 with various base models for invariant learning.

Base	Method	CIFAR10	CIFAR100	Food	MIT67	Pets	Flowers	Caltech101	Cars	Aircraft	DTD	SUN397	Mean
BYOL	-	85.55	59.80	37.54	42.61	50.61	73.50	72.46	23.02	31.71	44.95	31.63	50.31
	AugSelf	87.01	64.84	**43.14**	**47.24**	**52.49**	78.88	75.42	25.47	37.02	**48.03**	**34.94**	54.04
	EquiMod	84.64	56.55	32.74	39.18	44.64	66.54	68.37	15.47	24.27	42.71	26.96	45.64
	STL	**86.88**	**65.63**	42.98	46.42	52.33	**79.61**	**76.04**	**28.68**	**39.21**	46.44	34.57	**54.44**
SimSiam	-	83.26	55.69	34.32	40.52	46.52	66.06	69.13	17.15	27.99	41.91	28.97	46.50
	AugSelf	**85.44**	62.20	39.78	43.43	46.77	**77.90**	**71.72**	18.67	**33.30**	45.53	**32.65**	50.67
	EquiMod	81.20	51.23	31.21	37.99	40.53	63.98	64.19	12.22	22.11	40.69	25.76	42.83
	STL	85.20	**62.58**	**40.15**	**44.03**	**48.65**	76.68	71.37	**22.42**	32.37	**45.59**	32.19	**51.02**
Barlow Twins	-	81.67	51.68	27.79	33.13	39.60	57.63	62.17	11.53	19.47	37.13	23.43	40.48
	AugSelf	82.46	51.71	27.83	35.75	39.33	58.24	61.87	11.88	19.77	37.29	23.31	40.86
	EquiMod	81.57	52.15	30.00	36.79	38.70	62.64	63.22	11.80	20.55	40.21	24.92	42.05
	STL	**83.74**	**56.73**	**32.69**	**38.36**	**42.65**	**67.28**	**68.09**	**16.24**	**24.33**	**41.97**	**28.53**	**45.51**

Transformations. Table 8 shows STL's flexibility and effectiveness across various transformations. STL achieves the highest mean accuracy across all settings, outperforming AugSelf and EquiMod in both single transformations like crop and color and in combined transformations of crop and color or all transformations, reaching 49.87% and 48.48% mean accuracy, respectively. These results highlight STL's robust equivariant learning across diverse transformation types, enabling strong generalization without constraints on specific transformations.

Base Invariant Learning Models. Table 9 demonstrates STL's compatibility with various base models such as SimCLR, BYOL, SimSiam, and Barlow Twins. STL consistently improves the mean accuracy across all base models, outperforming both AugSelf and EquiMod, with the highest overall mean accuracy achieved with BYOL at 54.44%. These results demonstrate that STL is compatible with different invariant learning frameworks, confirming its adaptability and effectiveness regardless of the underlying representation learning method.

5 Related Works

Transformation equivariant learning captures transformation-sensitive features by embedding transformation directly into the learning process, categorized into implicit and explicit approaches. Implicit learning predicts transformations by observing changes in representations, allowing models to infer transformation without directly modeling the functions. In contrast, explicit equivariant learning encodes transformations within the representation space, enforcing behaviors in learned representations that mirror input transformations. These approaches can be reviewed in detail in the Appendix G.

Implicit Equivariant Learning. In implicit equivariant learning, models learn to recognize applied transformations by observing changes in representations, enabling the representations to capture transformation-sensitive information. Notable methods include InfoMin [43], selecting optimal views to maintain relevant task information, and Prelax [45], aligning residual vectors for robust multi-view alignment. Similarly, E-SSL [8] and AugSelf [28] leverage transformation-aware auxiliary tasks to train models to preserve transformation-sensitive details, enhancing robustness by maintaining sensitivity to transformation variance.

Explicit Equivariant Learning. On the other hand, explicit equivariant learning directly encodes input transformation into the representation space, building equivariant transformations that operate consistently in representation space. AEAE [19] leverages group actions to embed transformation effects explicitly in the representation space, while SymReg [40] enhances transformation consistency by selecting optimal loss terms based on transformation group information. CARE [20] introduces rotational symmetry by aligning embeddings directly with input rotations, providing robust transformation encoding. Similarly, SEN [36] applies symmetric embedding networks to synchronize transformations in the input space with learned representations. In a more flexible approach, EquiMod [9] models equivariant transformations by conditioning transformation labels as inputs and dynamically adapting representations through a neural network. Building on these approaches, SIE [13] separates invariant and equivariant representations, using dedicated networks to distinctly capture transformation-sensitive and invariant aspects.

Applications of Equivariant Learning. The applicability of equivariant learning extends across various domains, including robotics, medical imaging, molecular modeling, and multimodal representation learning. By leveraging inherent symmetries within data, it enhances sample efficiency in robotic manipulation [44], improves accuracy in medical image processing [21], and boosts predictive performance in molecular data analysis [46]. In multimodal contexts, it achieves finer alignment by considering transformations within each modality, facilitating robust cross-modal understanding [29, 14]. These applications underscore the broad utility of equivariant learning in aligning model representations with domain-specific transformations.

6 Discussion and Conclusion

The Self-supervised Transformation Learning (STL) approach introduces a novel method for deriving transformation representations that capture equivariance without relying on predefined transformation labels. This framework optimizes representational versatility through a synergy of three key loss functions: invariant learning, equivariant learning, and self-supervised transformation learning. These loss functions allow representations to adapt to complex transformation dynamics, including interdependencies among transformations, thereby greatly enhancing model generalization. STL consistently outperforms existing methods across 7 out of 11 classification tasks and demonstrates exceptional performance in object detection, proving its strong ability to generalize across diverse transformations. Additionally, integrating STL with AugMix yields robust performance improvements across all tasks, demonstrating enhanced resilience. STL's adaptability and consistent performance across various transformations and base models underscore its versatility for broad applications.

Limitations. While STL significantly advances equivariant learning, it encounters challenges with transformations that extend beyond single image pairs. Complex transformations, such as those involving combinations or mixtures of multiple images (e.g., mixup [49]), fall outside STL's current capacity as it relies on pairwise transformation representations. Further research could explore ways to adapt STL to accommodate more complex, multi-image transformations and better capture their inherent structure.

Broader Impacts. STL holds promise for applications requiring precise and interpretable transformations, such as in medical imaging analysis and autonomous driving. However, as STL learns transformation representations from the data, it may inherit biases embedded in the training data, raising fairness concerns, especially in sensitive domains. Implementing fairness-aware training techniques and thorough validation processes could help mitigate these risks. Additionally, while STL advances model robustness and generalization, its computational demands may have environmental implications. Efficiency improvements, such as model distillation, could reduce the model's energy footprint, supporting sustainable deployment.

Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2024-00439020, Developing Sustainable, Real-Time Generative AI for Multimodal Interaction, SW Starlab).

References

[1] A. Bardes, J. Ponce, and Y. LeCun. Variance-invariance-covariance regularization for self-supervised learning. *ICLR, Vicreg*, 1:2, 2022.

[2] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014.

[3] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924, 2020.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[5] X. Chen and K. He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

[6] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[7] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[8] R. Dangovski, L. Jing, C. Loh, S. Han, A. Srivastava, B. Cheung, P. Agrawal, and M. Soljačić. Equivariant contrastive learning. *arXiv preprint arXiv:2111.00899*, 2021.

[9] A. Devillers and M. Lefort. Equimod: An equivariance module to improve self-supervised learning. *arXiv preprint arXiv:2211.01244*, 2022.

[10] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[11] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[12] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[13] Q. Garrido, L. Najman, and Y. Lecun. Self-supervised learning of split invariant equivariant representations. *arXiv preprint arXiv:2302.10283*, 2023.

[14] T. Geng, T. Wang, Y. Zhang, J. Duan, W. Guan, and F. Zheng. Uniav: Unified audio-visual perception for multi-task video localization. *arXiv preprint arXiv:2404.03179*, 2024.

[15] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[16] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[17] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[18] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[19] X. Guo, E. Zhu, X. Liu, and J. Yin. Affine equivariant autoencoder. In *IJCAI*, pages 2413–2419, 2019.

[20] S. Gupta, J. Robinson, D. Lim, S. Villar, and S. Jegelka. Structuring representation geometry with rotationally equivariant contrastive learning. *arXiv preprint arXiv:2306.13924*, 2023.

[21] A. Hashemi, Y. Feng, and H. Sabet. Spherical cnn for medical imaging applications: Importance of equivariance in image reconstruction and denoising. *ArXiv*, 2023.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[23] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[24] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[25] S. Kornblith, J. Shlens, and Q. V. Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

[26] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[27] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[28] H. Lee, K. Lee, K. Lee, H. Lee, and J. Shin. Improving transferability of representations via augmentation-aware self-supervision. *Advances in Neural Information Processing Systems*, 34:17710–17722, 2021.

[29] J. Lee, J. Kim, H. Shon, B. Kim, S. H. Kim, H. Lee, and J. Kim. Uniclip: Unified framework for contrastive language-image pre-training. *Advances in Neural Information Processing Systems*, 35:1008–1019, 2022.

[30] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[31] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[32] L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[33] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pages 722–729. IEEE, 2008.

[34] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

[35] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[36] J. Y. Park, O. Biza, L. Zhao, J. W. van de Meent, and R. Walters. Learning symmetric embeddings for equivariant world models. *arXiv preprint arXiv:2204.11371*, 2022.

[37] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[38] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *2009 IEEE conference on computer vision and pattern recognition*, pages 413–420. IEEE, 2009.

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[40] M. Shakerinava, A. K. Mondal, and S. Ravanbakhsh. Structuring representations using group invariants. *Advances in Neural Information Processing Systems*, 35:34162–34174, 2022.

[41] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[42] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[43] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning? *Advances in neural information processing systems*, 33:6827–6839, 2020.

[44] D. Wang, M. Jia, X. Zhu, R. Walters, and R. Platt. On-robot learning with equivariant models. *arXiv preprint arXiv:2203.04923*, 2022.

[45] Y. Wang, Z. Geng, F. Jiang, C. Li, Y. Wang, J. Yang, and Z. Lin. Residual relaxation for multi-view representation learning. *Advances in Neural Information Processing Systems*, 34:12104–12115, 2021.

[46] R. Winter, M. Bertolini, T. Le, F. Noé, and D.-A. Clevert. Unsupervised learning of group invariant and equivariant representations. *Advances in Neural Information Processing Systems*, 35:31942–31956, 2022.

[47] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[48] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[49] H. Zhang. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

A STL Formulations for Various Base Invariant Models

A.1 STL Extension on BYOL

In adapting STL to BYOL [18], we utilize dissimilarity loss of BYOL to define the invariant, equivariant, and transformation losses. BYOL's dissimilarity metric is:

$$\mathcal{L}_{\text{BYOL}}(y, y^+; g, q, \theta, \xi) = \|\overline{q}_\theta(g_\theta(y)) - \overline{g}_\xi(y^+)\|_2^2, \tag{15}$$

where g_θ denotes the projection network parameterized by θ, q_θ is the prediction network also parameterized by θ, and g_ξ represents the target network parameterized by ξ. The terms $\overline{q}_\theta$ and $\overline{g}_\xi$ refer to the normalized outputs of q_θ and g_ξ, respectively. Using this, we define the STL objectives as follows:

$$\mathcal{L}_{\text{inv}}(x, t) = \mathcal{L}_{\text{BYOL}}(f(x), f(t(x)); g_{\text{inv}}, q_{\text{inv}}, \theta, \xi), \tag{16}$$

$$\mathcal{L}_{\text{equi}}(x, x', t) = \mathcal{L}_{\text{BYOL}}(\phi(y_t^{x'}, f(x)), f(t(x)); g_{\text{equi}}, q_{\text{equi}}, \theta, \xi), \tag{17}$$

$$\mathcal{L}_{\text{trans}}(x, x', t) = \mathcal{L}_{\text{BYOL}}(y_t^x, y_t^{x'}; g_{\text{trans}}, q_{\text{trans}}, \theta, \xi). \tag{18}$$

A.2 STL Extension on SimSiam

For SimSiam [5], STL uses SimSiam's dissimilarity loss:

$$\mathcal{L}_{\text{SimSiam}}(y, y^+; g, h) = \frac{1}{2}\mathcal{D}(h(g(y)), \text{stopgrad}(g(y^+))) + \frac{1}{2}\mathcal{D}(h(g(y^+)), \text{stopgrad}(g(y))), \tag{19}$$

where g denotes the projection network, h is the prediction network, stopgrad indicates an operation that halts gradient backpropagation, and $\mathcal{D}$ represents cosine similarity. This enables us to structure the STL losses as:

$$\mathcal{L}_{\text{inv}}(x, t) = \mathcal{L}_{\text{SimSiam}}(f(x), f(t(x)); g_{\text{inv}}, h_{\text{inv}}), \tag{20}$$

$$\mathcal{L}_{\text{equi}}(x, x', t) = \mathcal{L}_{\text{SimSiam}}(\phi(y_t^{x'}, f(x)), f(t(x)); g_{\text{equi}}, h_{\text{equi}}), \tag{21}$$

$$\mathcal{L}_{\text{trans}}(x, x', t) = \mathcal{L}_{\text{SimSiam}}(y_t^x, y_t^{x'}; g_{\text{trans}}, h_{\text{trans}}). \tag{22}$$

A.3 STL Extension on Barlow Twins

For Barlow Twins [48], STL applies Barlow Twins' dissimilarity loss:

$$\mathcal{L}_{\text{BarlowTwins}}(Y = \{y_i\}_i, Y^+ = \{y_i^+\}_i; g, \lambda) = \sum_i (1 - \mathcal{C}_{ii})^2 + \lambda \sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2, \tag{23}$$

where $\mathcal{C}$ denotes the cross-correlation matrix between embeddings of transformed views Y and Y^+, and λ is a regularization parameter that controls the weight of off-diagonal terms in $\mathcal{C}$, penalizing redundancy in the representations. Define $X = \{x_i\}_i$, $X' = \{x_i'\}_i$, and $T = \{t_i\}_i$, representing the sets of input images, paired images, and transformations, respectively. Then, the STL losses for Barlow Twins are:

$$\mathcal{L}_{\text{inv}}(X, T) = \mathcal{L}_{\text{BarlowTwins}}(\{f(x_i)\}_i, \{f(t_i(x_i))\}_i; g_{\text{inv}}, \lambda_{\text{inv}}), \tag{24}$$

$$\mathcal{L}_{\text{equi}}(X, X', T) = \mathcal{L}_{\text{BarlowTwins}}(\{\phi(y_{t_i}^{x_i'}, f(x_i))\}_i, \{f(t_i(x_i))\}_i; g_{\text{equi}}, \lambda_{\text{equi}}), \tag{25}$$

$$\mathcal{L}_{\text{trans}}(X, X', T) = \mathcal{L}_{\text{BarlowTwins}}(\{y_{t_i}^{x_i}\}_i, \{y_{t_i}^{x_i'}\}_i; g_{\text{trans}}, \lambda_{\text{trans}}). \tag{26}$$

B Datasets

Table 10: **Dataset Information.** Overview of dataset composition and evaluation metrics. Each dataset specifies the number of classes, training/validation/test splits, and the corresponding evaluation metric.

Category	Dataset	# of classes	Training	Validation	Test	Metric
(a) Pretraining	STL10 [7]	10	105,000	-	-	-
	ImageNet100 [39, 42]	1,000	126,689	-	-	-
(b) Linear Evaluation	CIFAR10 [27]	10	45,000	5,000	10,000	Top-1 accuracy
	CIFAR100 [27]	100	45,000	5,000	10,000	Top-1 accuracy
	Food [2]	101	68,175	7,575	25,250	Top-1 accuracy
	MIT67 [38]	67	4,690	670	1,340	Top-1 accuracy
	Pets [37]	37	2,940	740	3,669	Mean per-class accuracy
	Flowers [33]	102	1,020	1,020	6,149	Mean per-class accuracy
	Caltech101 [12]	101	2,525	505	5,647	Mean Per-class accuracy
	Cars [26]	196	6,494	1,650	8,041	Top-1 accuracy
	Aircraft [31]	100	3,334	3,333	3,333	Mean Per-class accuracy
	DTD (split 1) [6]	47	1,880	1,880	1,880	Top-1 accuracy
	SUN397 (split 1) [47]	397	15,880	3,970	19,850	Top-1 accuracy
(c) Object Detection	VOC2007+2012 [11]	20	16,551	-	4,952	Average Precision

Table 10 presents detailed descriptions of (a) pre-taining dataset, (b) linear evaluation datasets, and (c) object detection and instance segmentation dataset. For linear evaluation dataset, validation samples are randomly selected from the training split if an official validation split is not provided. For the object detection and instance segmentation dataset, we use `trainval` set for training VOC07+12 [11] while only using `test` set for testing.

C Evaluation Protocols

Linear Evaluation. Following established protocols [25, 4, 18], we train linear classifiers on frozen representations from center-cropped images resized to 224×224 (or 96×96 for STL10). No data augmentation is applied. Each image is resized along its shorter side to 224 and then center-cropped to 224×224. The ℓ_2-regularized cross-entropy objective is minimized using L-BFGS, with regularization selected from 45 logarithmically spaced values between 10^{-6} and 10^5 on the validation set. The optimal model is then retrained on both training and validation splits, with test accuracy reported. L-BFGS is capped at 5,000 iterations, with each step initialized from the previous solution.

Object Detection. We use the VOC2007+2012 `trainval` set with 16,551 images to train a Faster R-CNN [16] with a C-4 backbone. Training spans 24,000 iterations with a batch size of 16 and SyncBatchNorm. The learning rate starts at 0.1, decreasing by a factor of 10 at 18,000 and 22,000 iterations. A linear warmup [17] is applied for the first 1,000 iterations with a 0.333 slope.

D Transformation Labels

In this section, we describe the labels of the transformations utilized in AugSelf [28] (random crop, horizontal flip, color jitter, grayscale, and Gaussian blur) in the following. The labels are designed by the parameters in each transformation.

- `RandomResizedCrop`. The labels are constructed with the center points, H_{center} and W_{center}, of the crop and the height H and width W values of the cropping area.
- `RandomHorizontalFlip`. As flip transformation is a simple operation, the label is 0 or 1.
- `ColorJitter`. The four parameters of color jitter are brightness, contrast, saturation, and hue. Each parameter has its own range, with brightness, contrast, and saturation ranging from 0.0 to 1.0, while hue ranges from 0.0 to 0.5.
- `RandomGrayscale`. This transformation applies the grayscale to the image. In line with the flip, the label is 0 or 1.
- `GaussianBlur`. The standard deviation is utilized for the Gaussian blur label ranging from 0.1 to 2.0.

E Pretraining Setups

For the pretraining experiments, we use NVIDIA RTX4090.

E.1 ImageNet100 Pretraining

We conduct pretraining on the ResNet-50 architecture [22] using ImageNet100, a subset of ImageNet containing 100 categories [39], with dataset splits consistent with those in [42]. All methods are trained for 500 epochs with a batch size of 256, using a cosine learning rate schedule without restarts [30]. The initial learning rate is set at 0.03, with a weight decay of 0.0005. The model includes a 3-layer projection MLP head, $g(\cdot)$, with a hidden dimension of 2048 and an output dimension of 128. Batch normalization [24] is excluded from the last layer.

E.2 STL10 Pretraining

For pretraining on the STL10 dataset [7], we use the standard ResNet-18 architecture [22]. All methods utilize stochastic gradient descent (SGD) with a learning rate of 0.03, a batch size of 256, a weight decay of 0.0005, and a momentum of 0.9. The learning rate follows a cosine decay schedule without restarts [30].

SimCLR [4]. A 3-layer projection MLP head $g(\cdot)$ with a hidden dimension of 512 and an output dimension of 128 is used, with batch normalization excluded from the final layer. In contrastive learning, we apply a temperature scaling parameter of 0.2.

Barlow Twins [48]. A 2-layer projection MLP head $g(\cdot)$ is employed, with a hidden dimension of 512 and an output dimension of 2048. Batch normalization is excluded from the last layer.

BYOL [18]. The model uses a 2-layer projection MLP head $g(\cdot)$, with a hidden dimension of 4096 and an output dimension of 256, omitting batch normalization in the final layer.

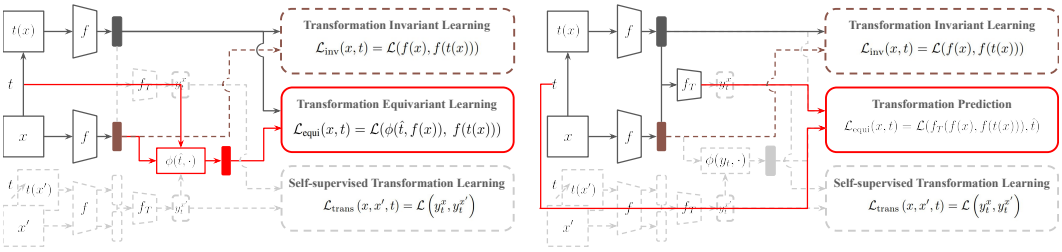
SimSiam [5]. We employ a 2-layer projection MLP head $g(\cdot)$ with both hidden and output dimensions of 2048, with batch normalization excluded from the final layer.

F Image Classification Results of STL10-pretrained Models

Table 11: **Image Classification.** Evaluation of representation on in-domain and 11 downstream out-domain classification task: Linear evaluation accuracy (%) of ResNet-18 [22] pretrained on STL10 [7] averaged over three random seeds (mean ± std).

Method	In-domain	Out-domain											
	STL10	CIFAR10	CIFAR100	Food	MIT67	Pets	Flowers	Caltech101	Cars	Aircraft	DTD	SUN397	Mean
Transformation Invariant Learning :													
SimCLR	84.74±0.18	80.89±2.70	51.12±2.50	32.23±0.18	37.61±1.13	44.10±0.38	63.55±0.67	66.17±0.55	14.44±0.21	24.13±0.84	40.32±0.33	26.23±0.10	43.72±0.65
with AugMix	**85.58±0.16**	81.72±0.75	52.42±0.36	33.22±0.31	39.63±0.98	45.16±0.78	65.32±0.68	69.81±0.30	15.66±0.74	25.79±1.00	42.06±0.55	27.95±0.06	45.34±0.21
Implicit Transformation Equivariant Learning :													
E-SSL	85.19±0.08	82.82±2.17	54.89±2.98	35.11±0.16	39.55±0.20	45.06±0.15	69.19±1.47	68.23±0.65	16.51±0.23	26.84±0.80	43.87±0.82	28.80±0.06	46.44±0.33
AugSelf	84.99±1.05	84.12±0.94	57.59±0.80	36.63±0.08	40.90±0.97	46.09±0.44	72.45±1.53	69.58±0.27	17.58±0.17	27.73±0.43	44.24±1.68	30.49±0.35	47.94±0.25
Explicit Transformation Equivariant Learning :													
SEN	78.66±0.40	81.01±0.57	51.59±1.10	30.00±0.23	34.00±1.26	35.76±0.28	60.93±1.18	64.35±0.16	12.12±0.51	24.87±1.32	38.12±0.31	23.24±0.30	41.45±0.32
EquiMod	84.28±0.18	83.63±1.59	55.94±2.34	34.01±0.49	38.78±0.63	42.94±0.86	66.75±0.69	68.70±0.87	15.49±0.72	26.84±1.11	43.03±1.05	27.41±0.43	45.77±0.79
SIE	83.60±0.13	82.60±1.89	53.43±2.59	32.97±0.27	37.14±0.99	41.03±0.25	65.24±1.07	67.07±0.29	14.04±0.54	25.91±0.68	42.04±0.35	25.97±0.12	44.31±0.64
STL (Ours)	84.83±0.21	85.22±0.17	60.13±0.81	38.05±1.19	43.53±2.04	46.57±0.56	73.50±2.58	71.36±0.20	18.85±0.17	30.25±1.00	45.34±0.73	31.63±0.44	49.49±0.42
with AugMix	**85.57±0.19**	**86.01±0.47**	**62.07±0.25**	**40.16±0.13**	**44.90±1.00**	**46.69±0.23**	**77.37±0.79**	**73.29±0.19**	**19.32±0.48**	**30.87±1.04**	**48.71±0.67**	**33.44±0.20**	**51.17±0.12**

G Explicit and Implicit Equivariant Learning



(a) Explicit Equivariant Learning (b) Implicit Equivariant Learning

Figure 5: **Explicit and Implicit Equivariant Learning.** Transformation equivariant learning with transformation labels is divided into (Left) explicit and (Right) implicit equivariant learning.

Figure 5 illustrates the framework of explicit and implicit equivariant learning. Explicit methods like SEN [36], EquiMod [9], and SIE [13] apply a direct equivariant transformation network to representations, leveraging transformation labels for alignment. In contrast, implicit methods, such as E-SSL [8] and AugSelf [28], infer transformation effects without directly applying transformations by utilizing auxiliary tasks to deduce transformation states.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The claim in the abstract and introduction is dealt with in the background, method, and experiment sections.

   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: We deal with this in the Discussion and 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: We provide the hyperparameters, training, and evaluation protocols with details in the paper and supplementary material.

   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: [No]

Justification: All training and evaluation are conducted based on publicly available codes, and we provide the details of our implementations.

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: We report all the details in the paper and supplementary material.

   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: [No]

   Justification: The experiment merely contains negligible statistical effects.

   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: We describe detailed information about the experiment in the paper and supplementary material.

   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: We wrote a paper in compliance with the code of ethics.

   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: [Yes]

    Justification: We denote the negative societal impacts that the proposed method may have.

    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: [NA]

    Justification: [NA]

    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: [NA]

    Justification: [NA]

    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.