南京航空航天大學

Nanjing University of Aeronautics and Astronautics

What Makes for Good Views for Contrastive Learning?

Yonglong Tian
MIT CSAIL

Chen Sun
Google, Brown University

Ben Poole
Google Research

Dilip Krishnan
Google Research

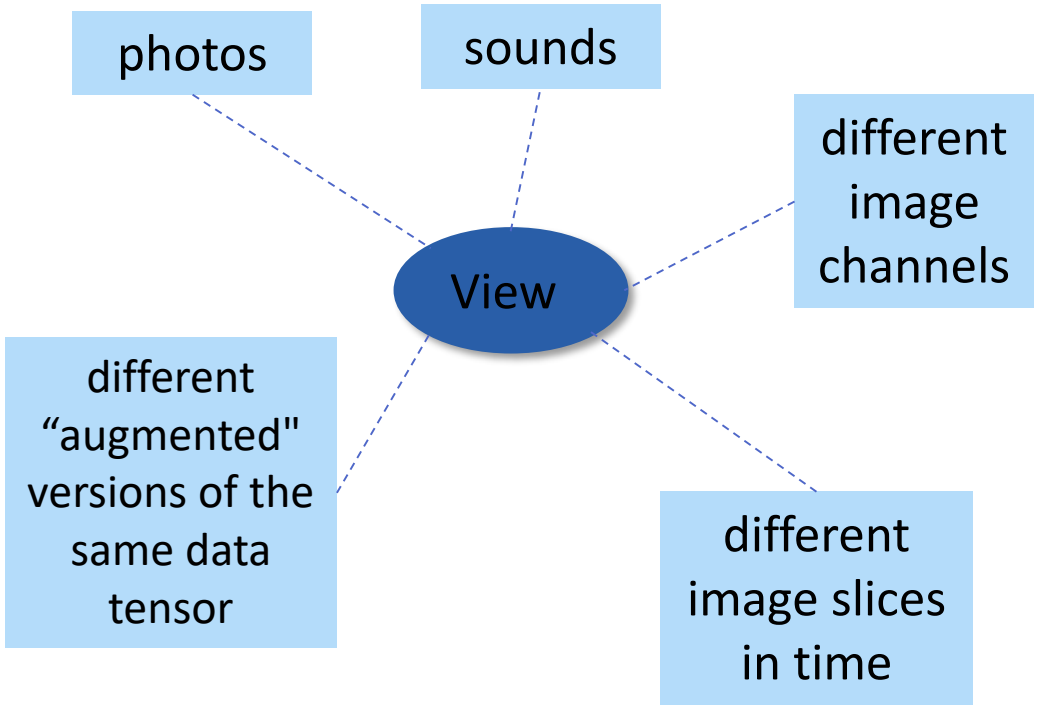
Cordelia Schmid
Google Research

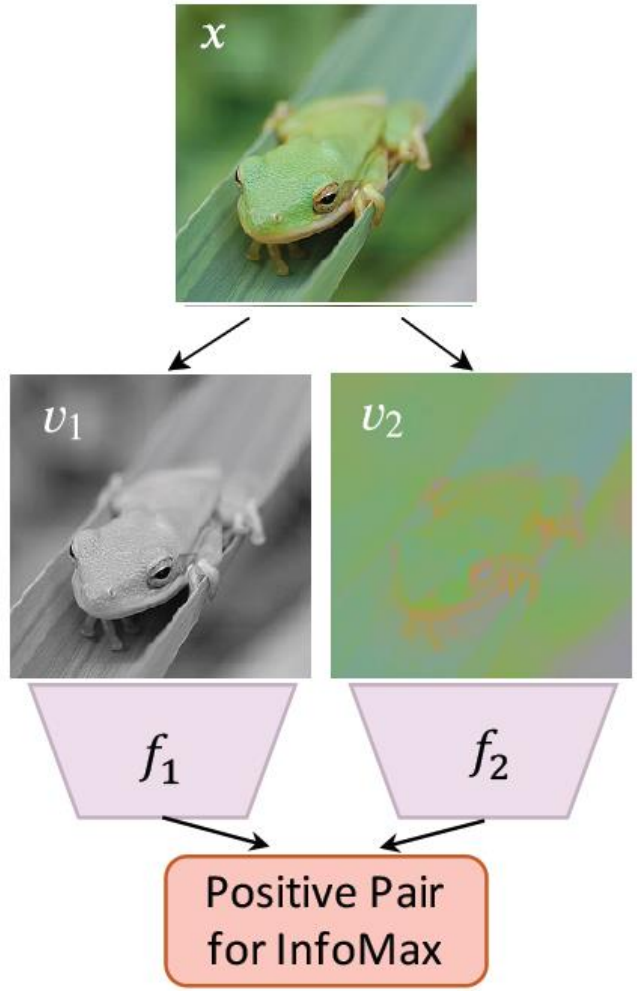
Phillip Isola
MIT CSAIL

NIPS 2020

Motivation

➢ Multi-View Contrastive Learning





Motivation

How can we find the right balance of views that share just the information we need, no more and no less?

Motivation

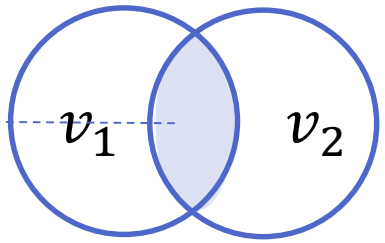
How can we find the right balance of views that share just the information we need, no more and no less?

⇩

1) The optimal choice of views depends critically on the downstream task.

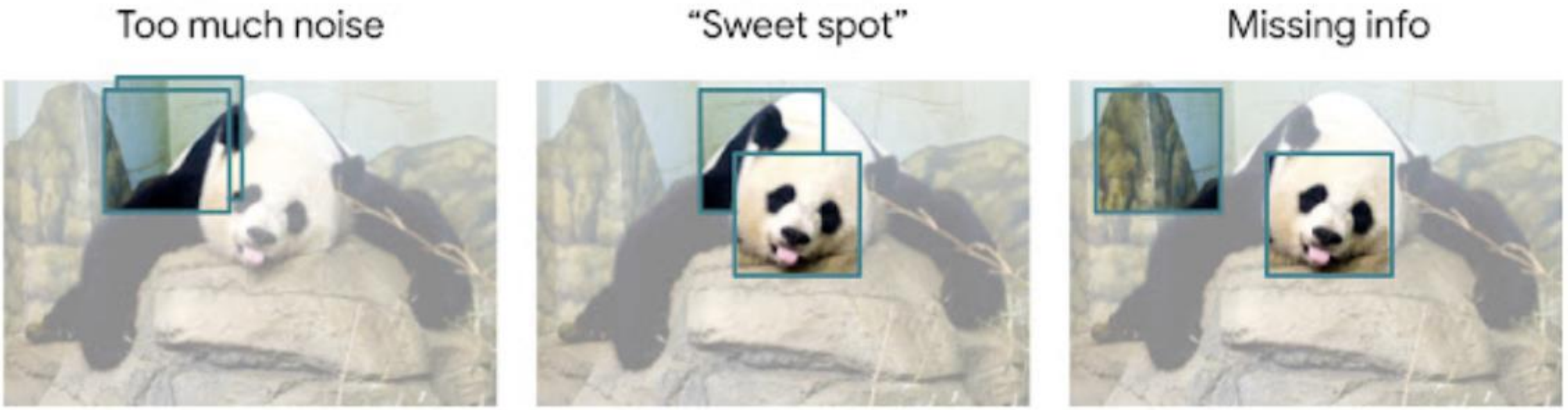
2) For many common ways of generating views, there is a sweet spot in terms of downstream performance where the mutual information (MI) between views is neither too high nor too low.

mutual information(MI) ------ $I(v_1; v_2)$ ----------



InfoMin principle

InfoMin principle: A good set of views are those that share the minimal information necessary to perform well at the downstream task.



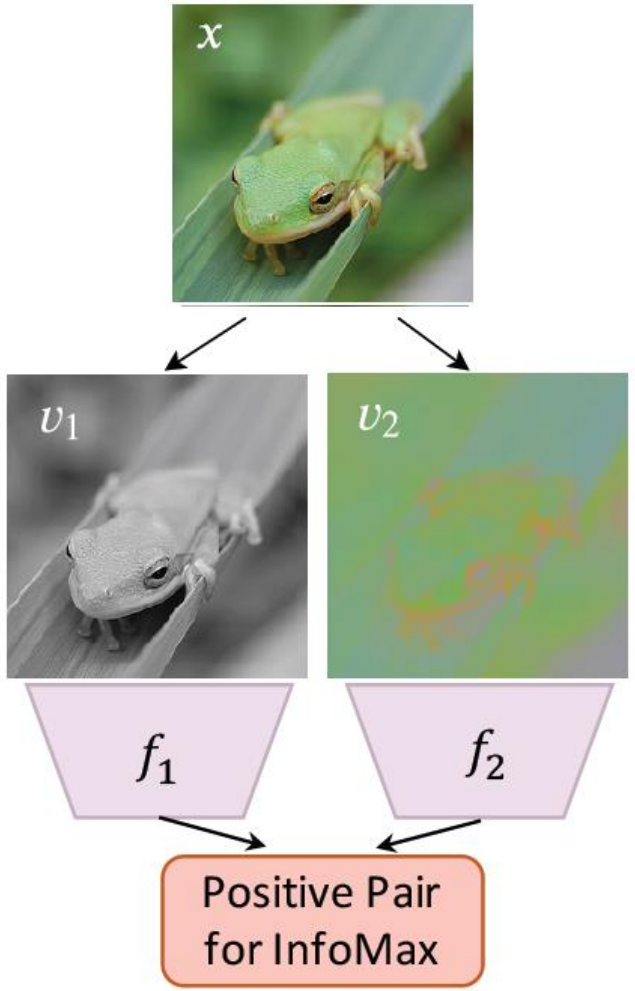
Formulation

InfoNCE loss:

$$\mathcal{L}_{\text{NCE}} = -\mathbb{E}\left[\log \frac{e^{h(\mathbf{v_{1,i}}, \mathbf{v_{2,i}})}}{\sum_{j=1}^{K} e^{h(\mathbf{v_{1,i}}, \mathbf{v_{2,j}})}}\right]$$

The score function $h(\cdot,\cdot)$ typically consists of two encoders (f_1 for v_1 and f_2 for v_2).

The resulting representations are $z_1 = f_1(v_1)$ and $z_2 = f_2(v_2)$.

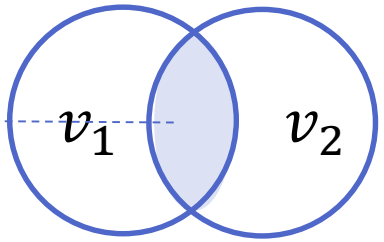


Formulation

Definition 1. *(Sufficient Encoder) The encoder* f_1 *of* $\mathbf{v_1}$ *is sufficient in the contrastive learning framework if and only if* $I(\mathbf{v_1}; \mathbf{v_2}) = I(f_1(\mathbf{v_1}); \mathbf{v_2})$.

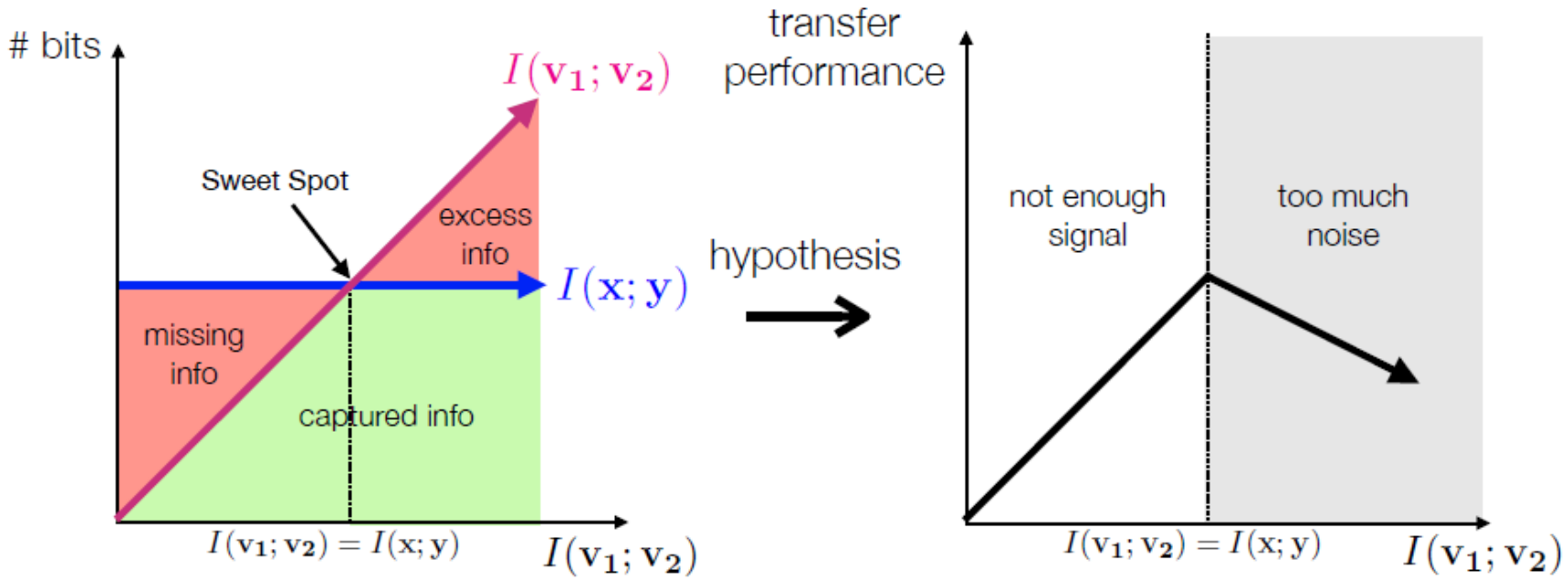
Definition 2. *(Minimal Sufficient Encoder) A sufficient encoder* f_1 *of* $\mathbf{v_1}$ *is minimal if and only if* $I(f_1(\mathbf{v_1}); \mathbf{v_1}) \leq I(f(\mathbf{v_1}); \mathbf{v_1}), \forall\ f$ *that is sufficient.*

Definition 3. *(Optimal Representation of a Task) For a task* $\mathcal{T}$ *whose goal is to predict a semantic label* $\mathbf{y}$ *from the input data* $\mathbf{x}$*, the optimal representation* $\mathbf{z}^*$ *encoded from* $\mathbf{x}$ *is the minimal sufficient statistic with respect to* $\mathbf{y}$.

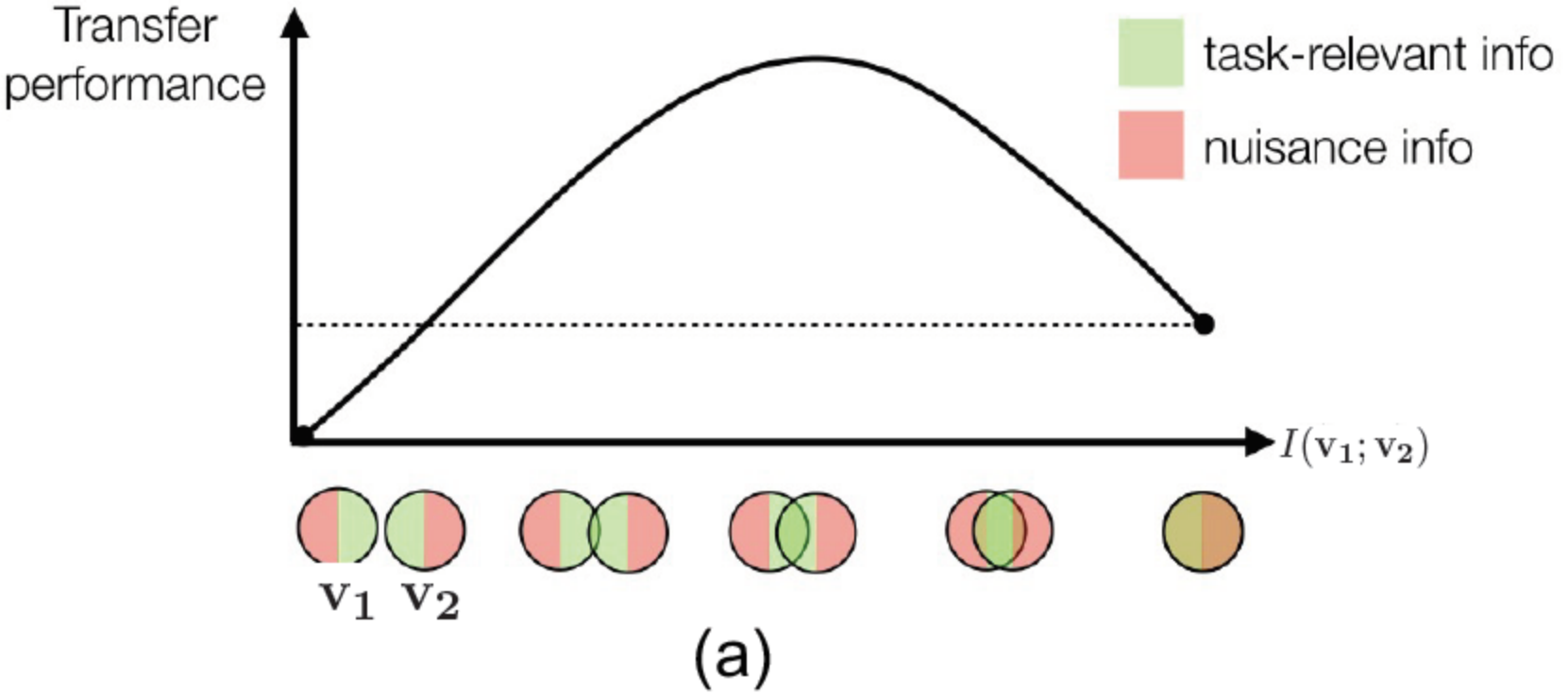
mutual information(MI) ------ $I(v_1; v_2)$ ---------- 

Formulation

1. *Missing information*: When $I(\mathbf{v_1}; \mathbf{v_2}) < I(\mathbf{x}; \mathbf{y})$, there is information about the task-relevant variable that is discarded by the view, degrading performance.
2. *Sweet spot*: When $I(\mathbf{v_1}; \mathbf{y}) = I(\mathbf{v_2}; \mathbf{y}) = I(\mathbf{v_1}; \mathbf{v_2}) = I(\mathbf{x}; \mathbf{y})$, the only information shared between $\mathbf{v_1}$ and $\mathbf{v_2}$ is task-relevant, and there is no irrelevant noise.
3. *Excess noise*: As we increase the amount of information shared in the views beyond $I(\mathbf{x}; \mathbf{y})$, we begin to include additional information that is irrelevant for the downstream task. This can lead to worse generalization on the downstream task [2, 58].



Theoretical analysis



Experiment

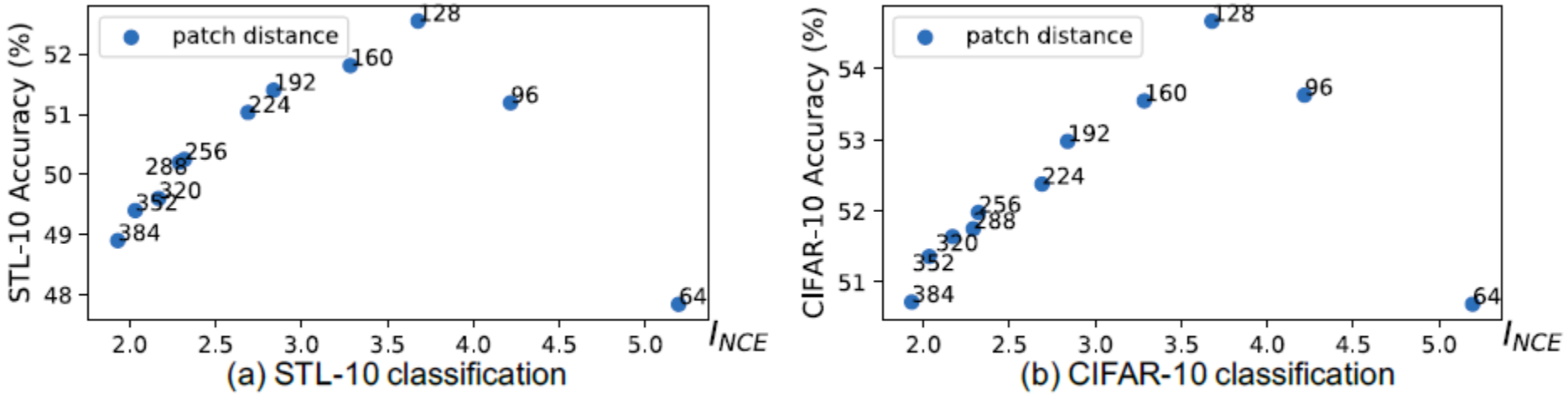


Figure 3: We create views by using pairs of image patches at various offsets from each other. As I_{NCE} is reduced, the downstream task accuracy firstly increases and then decreases, leading to a reverse-U shape.

Experiment

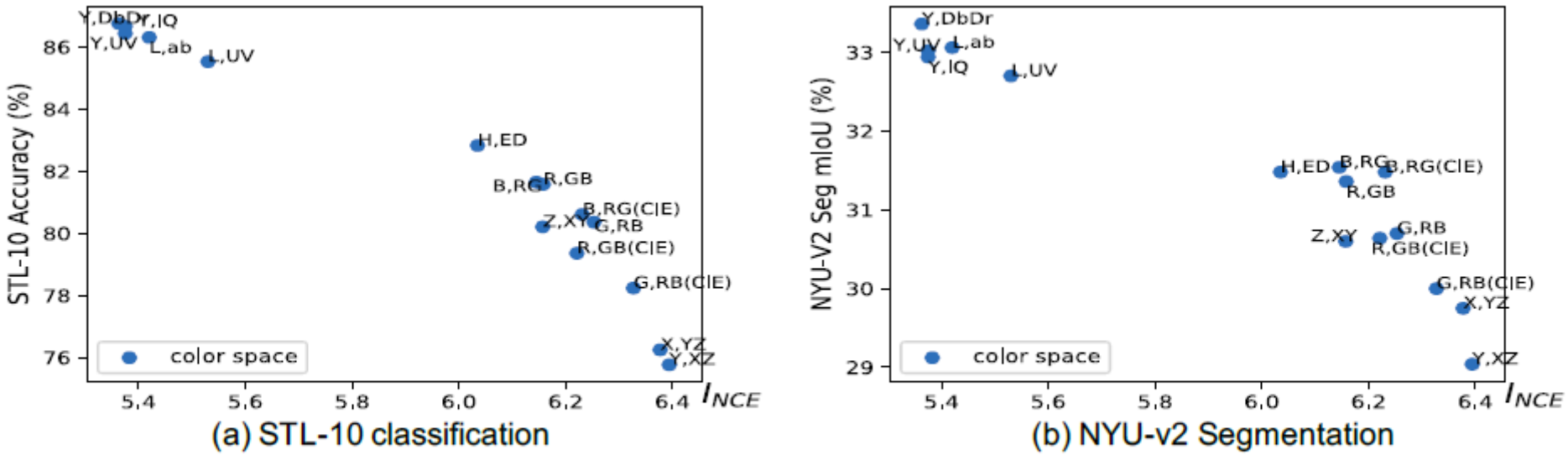


Figure 4: We build views by splitting channels of different color spaces. As I_{NCE} decreases, the accuracy on downstream tasks (STL-10 classification, NYU-v2 segmentation) improves.

Experiment

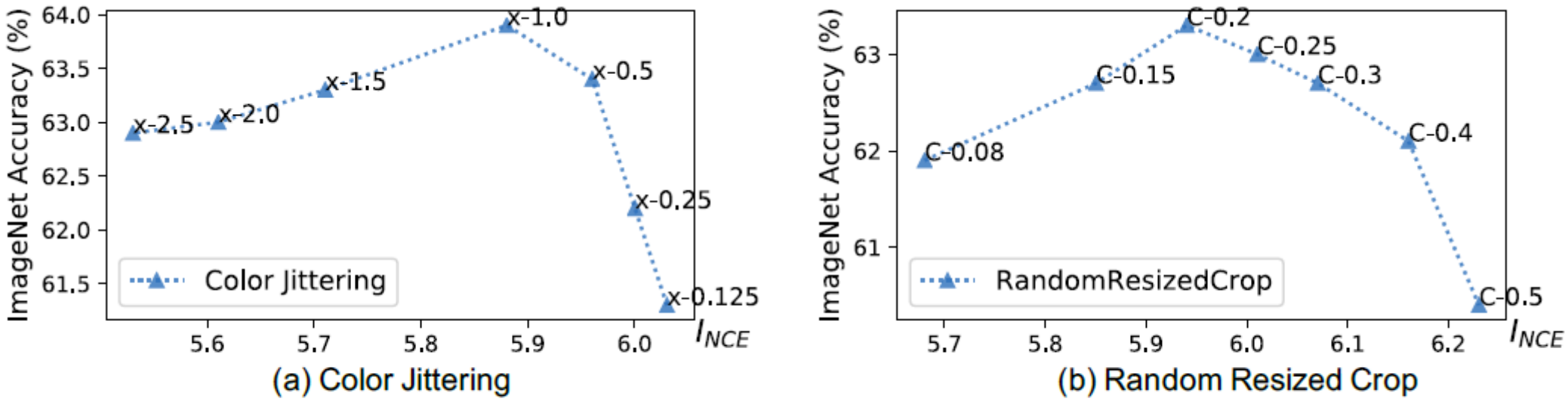
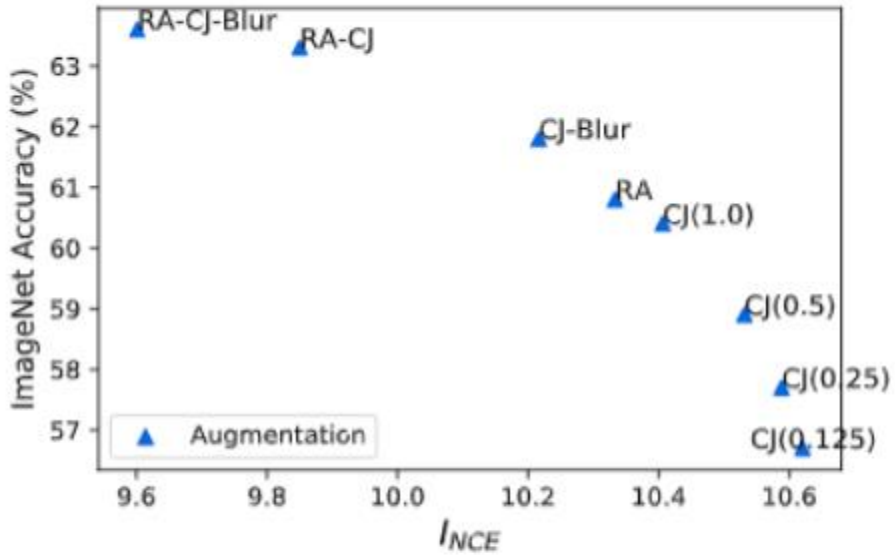


Figure 5: The reverse U-shape traced out by parameters of individual augmentation functions.

Proposed method: InfoMin Aug

(2) InfoMin Data Augmenation on ImageNet



(a) I_{NCE} *v.s* Accuracy

PyTorch-style data augmentation

```
RandomResizedCrop(scale=(0.2, 1.0))
RandomHorizontalFlip()
# CJ(x): random color jitter with x
cj = ColorJitter([0.8,0.8,0.8,0.4]*x)
RandomApply([cj], p=0.8)
# Blur: random blurring
blur = Blur(sigma=(0.1,2.0))
RandomApply([blur], p=0.5)
# RA: RandAugment
rnd_augment()
RandomGrayscale(p=0.2),
```

(b) Data Augmentation

Figure 3: The augmentation that we manually designed following the principle of InfoMin. As can be see from the left figure, lower I_{NCE} typically results in higher accuracy before we touch a turning point (which we might haven't touched yet).

Experiment

Table 1: Single-crop ImageNet accuracies (%) of linear classifiers [77] trained on representations learned with different contrastive methods using ResNet-50 [28]. InfoMin Aug. refers to data augmentation using *RandomResizedCrop*, *Color Jittering*, *Gaussian Blur*, *RandAugment*, *Color Dropping*, and a *JigSaw* branch as in PIRL [47]. * indicates splitting the network into two halves.

Method	Architecture	Param.	Head	Epochs	Top-1	Top-5
InstDis [70]	ResNet-50	24	Linear	200	56.5	-
Local Agg. [81]	ResNet-50	24	Linear	200	58.8	-
CMC [64]	ResNet-50*	12	Linear	240	60.0	82.3
MoCo [26]	ResNet-50	24	Linear	200	60.6	-
PIRL [47]	ResNet-50	24	Linear	800	63.6	-
CPC v2 [29]	ResNet-50	24	-	-	63.8	85.3
SimCLR [8]	ResNet-50	24	MLP	1000	69.3	89.0
InfoMin Aug. (Ours)	ResNet-50	24	MLP	200	70.1	89.4
InfoMin Aug. (Ours)	ResNet-50	24	MLP	800	**73.0**	**91.1**

Experiment

Table 2: Results of object detection and instance segmentation fine-tuned on COCO. We adopt Mask R-CNN **R50-FPN**, and report the bounding box AP and mask AP on `val2017`. In the brackets are the gaps to the ImageNet supervised pre-training counterpart. For fair comparison, InstDis [70], PIRL [47], MoCo [26], and InfoMin are all pre-trained for **200** epochs.

(a) Mask R-CNN, R50-FPN, **1x** schedule

pre-train	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
random init	32.8	50.9	35.3	29.9	47.9	32.0
supervised	39.7	59.5	43.3	35.9	56.6	38.6
InstDis [70]	38.8(↓0.9)	58.4(↓1.1)	42.5(↓0.8)	35.2(↓0.7)	55.8(↓0.8)	37.8(↓0.8)
PIRL [47]	38.6(↓1.1)	58.2(↓1.3)	42.1(↓1.2)	35.1(↓0.8)	55.5(↓1.1)	37.7(↓0.9)
MoCo [26]	39.4(↓0.3)	59.1(↓0.4)	42.9(↓0.4)	35.6(↓0.3)	56.2(↓0.4)	38.0(↓0.6)
MoCo v2 [9]	40.1(↑0.4)	59.8(↑0.3)	44.1(↑0.8)	36.3(↑0.4)	56.9(↑0.3)	39.1(↑0.5)
InfoMin Aug.	40.6(↑0.9)	60.6(↑1.1)	44.6(↑1.3)	36.7(↑0.8)	57.7(↑1.1)	39.4(↑0.8)

(b) Mask R-CNN, R50-FPN, **2x** schedule

pre-train	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
random init	38.4	57.5	42.0	34.7	54.8	37.2
supervised	41.6	61.7	45.3	37.6	58.7	40.4
InstDis [70]	41.3(↓0.3)	61.0(↓0.7)	45.3(↓0.0)	37.3(↓0.3)	58.3(↓0.4)	39.9(↓0.5)
PIRL [47]	41.2(↓0.4)	61.2(↓0.5)	45.2(↓0.1)	37.4(↓0.2)	58.5(↓0.2)	40.3(↓0.1)
MoCo [26]	41.7(↑0.1)	61.4(↓0.3)	45.7(↑0.4)	37.5(↓0.1)	58.6(↓0.1)	40.5(↑0.1)
MoCo v2 [9]	41.7(↑0.1)	61.6(↓0.1)	45.6(↑0.3)	37.6(↓0.0)	58.7(↓0.0)	40.5(↑0.1)
InfoMin Aug.	42.5(↑0.9)	62.7(↑1.0)	46.8(↑1.5)	38.4(↑0.8)	59.7(↑1.0)	41.4(↑1.0)

Experiment

Table 3: Pascal VOC object detection. All contrastive models are pretrained for **200** epochs on ImageNet for fair comparison. We use Faster R-CNN R50-**C4** architecture for object detection. APs are reported using the average of 5 runs. * we use numbers from [26] since the setting is exactly the same.

pre-train	AP_{50}	AP	AP_{75}	ImageNet Acc(%)
random init.*	60.2	33.8	33.1	-
supervised*	81.3	53.5	58.8	76.1
InstDis	80.9	55.2	61.2	59.5
PIRL	81.0	55.5	61.3	61.7
MoCo*	81.5	55.9	62.6	60.6
InfoMin Aug. (ours)	**82.7**	**57.6**	**64.6**	**70.1**

Learning views for contrastive learning

To understand how the choice of views impact the representations learned by contrastive learning, we construct a toy dataset that mixes three tasks.

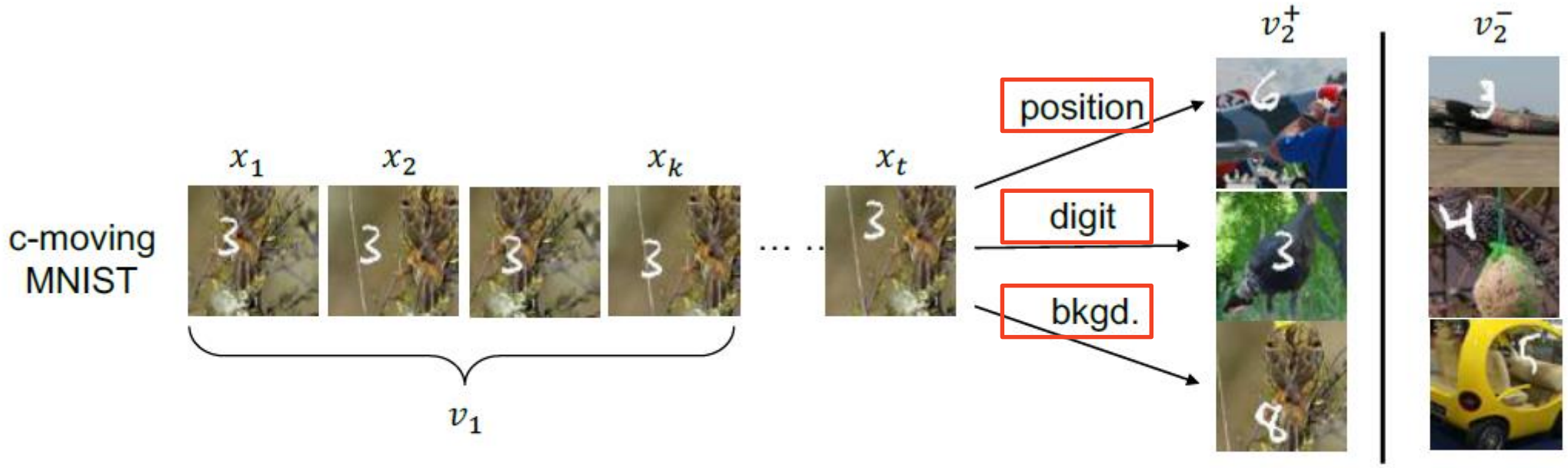


Figure 6: Illustration of the Colorful-Moving-MNIST dataset. In this example, the first view $\mathbf{v_1}$ is a sequence of frames containing the moving digit, e.g., $\mathbf{v_1} = x_{1:k}$. The matched second view $\mathbf{v_2^+}$ share some factor with x_t that $\mathbf{v_1}$ can predict, while the unmatched view $\mathbf{v_2^-}$ does not share factor with x_t.

Experiment

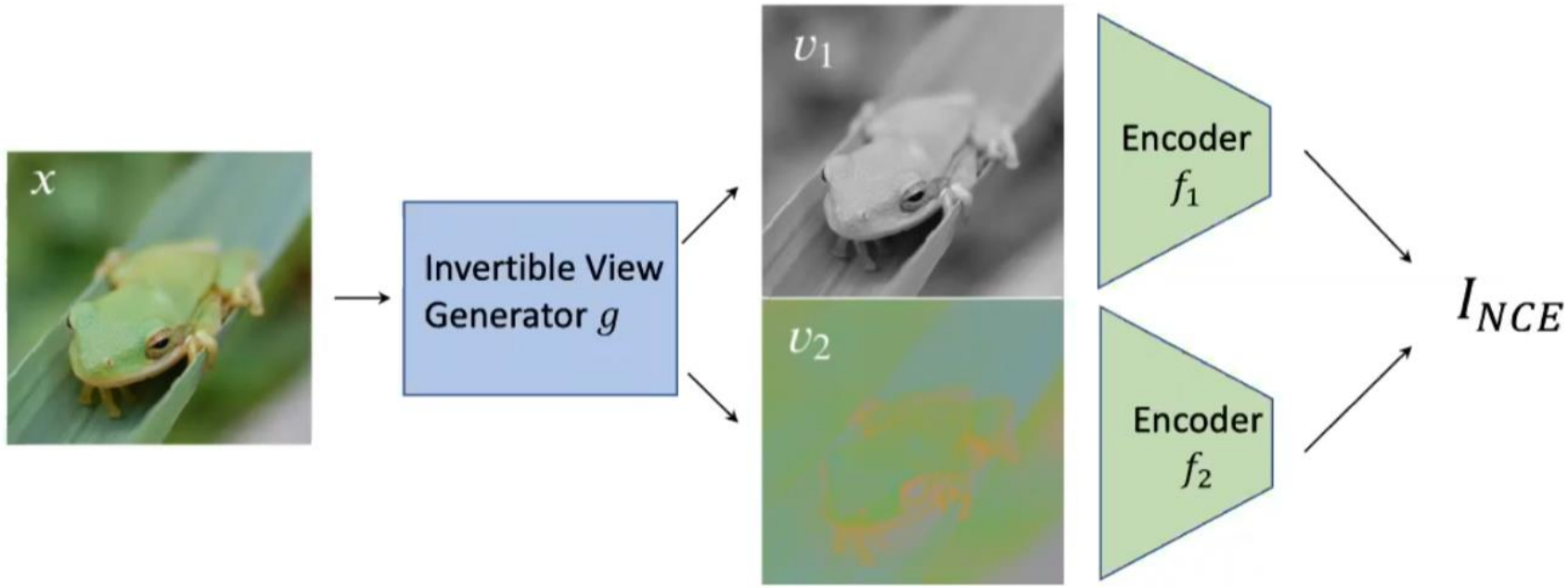
Table 4: We study how information shared by views $I(\mathbf{v_1}; \mathbf{v_2})$ would affect the representation quality, by evaluating on three downstream tasks: digit classification, localization, and background (STL-10) classification. Evaluation for contrastive methods is performed by freezing the backbone and training a linear task-specific head

	$I(\mathbf{v_1}; \mathbf{v_2})$	digit cls. error rate (%)	background cls. error rate (%)	digit loc. error pixels
Single Factor	*digit*	**16.8**	88.6	13.6
	bkgd	88.6	**51.7**	16.1
	pos	57.9	87.6	**3.95**
Multiple Factors	*bkgd, digit, pos*	88.8	56.3	16.2
	bkgd, digit	88.2	53.9	16.3
	bkgd, pos	88.8	53.8	15.9
	digit, pos	**14.5**	88.9	13.7
Supervised		3.4	45.3	0.93

Proposed method

- Unsupervised View Learning

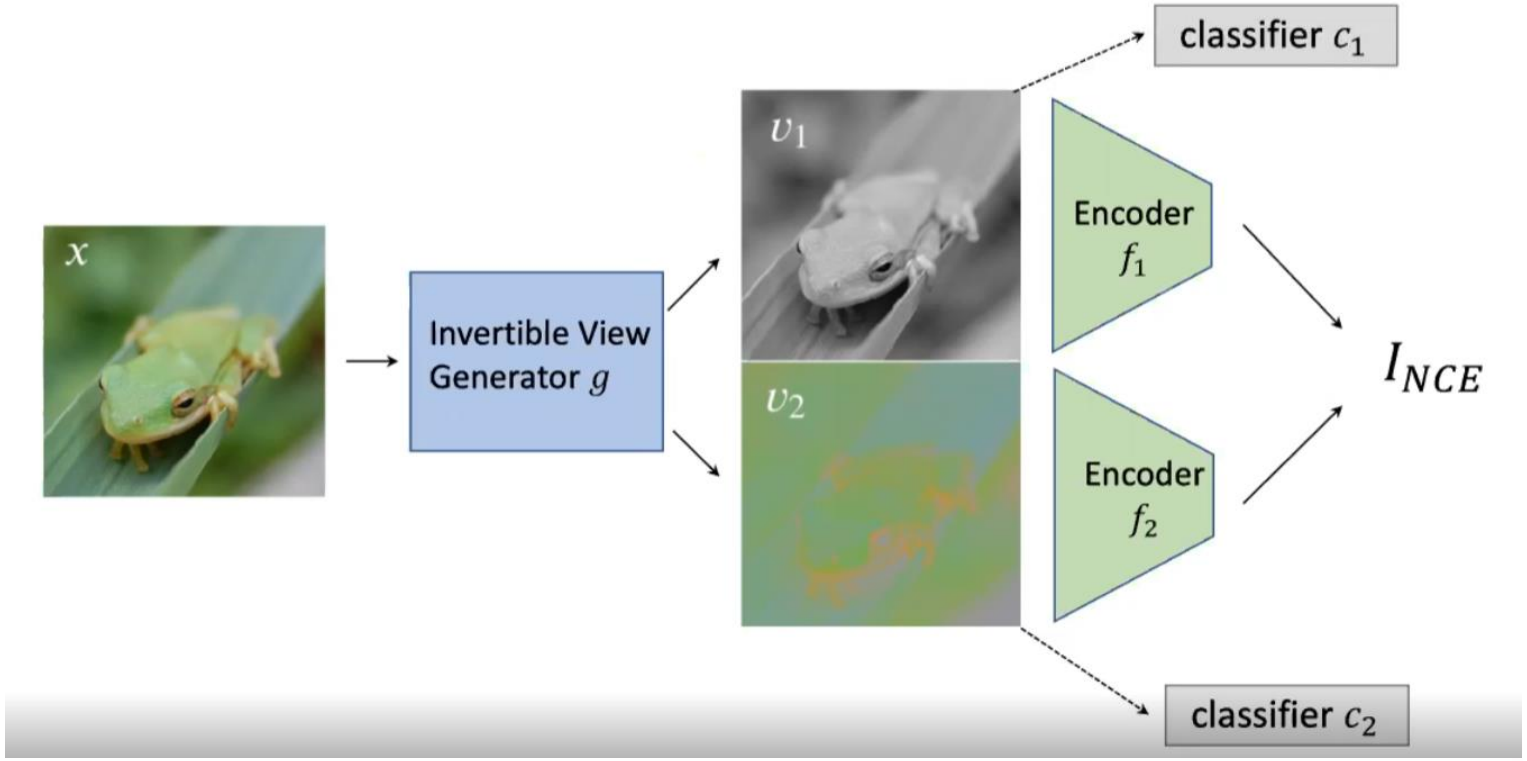
$$\min_{g} \max_{f_1, f_2} I_{\mathrm{NCE}}^{f_1, f_2}\left(g(X)_1; g(X)_{2:3}\right)$$



Proposed method

- Semi-supervised View Learning

$$\min_{g,c_1,c_2} \max_{f_1,f_2} \underbrace{I_{\mathrm{NCE}}^{f_1,f_2}(g(X)_1; g(X)_{2:3})}_{\text{unsupervised: reduce } I(v_1;v_2)} + \underbrace{\mathcal{L}_{ce}(c_1(g(X)_1), y) + \mathcal{L}_{ce}(c_2(g(X)_{2:3}), y)}_{\text{supervised: keep } I(v_1;y) \text{ and } I(v_2;y)}$$



Experiment

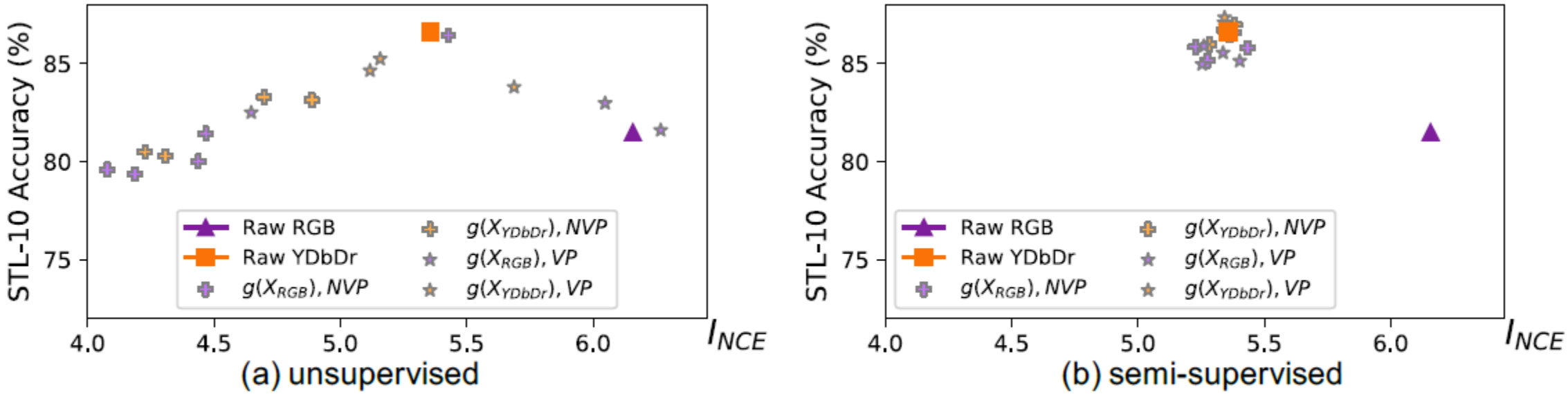


Figure 7: View generator learned by (a) unsupervised or (b) semi-supervised objectives.

Experiment

Table 5: Comparison of different view generators by measuring STL-10 classification accuracy: *supervised*, *unsupervised*, and *semi-supervised*. "# of Images" indicates how many images are used to learn view generators. In representation learning stage, all 105k images are used.

Method (# of Images)	RGB	YDbDr
unsupervised (100k)	82.4 ± 3.2	84.3 ± 0.5
supervised (5k)	79.9 ± 1.5	78.5 ± 2.3
semi-supervised (105k)	$\mathbf{86.0 \pm 0.6}$	$\mathbf{87.0 \pm 0.3}$
raw views	81.5 ± 0.2	86.6 ± 0.2

THANKS