



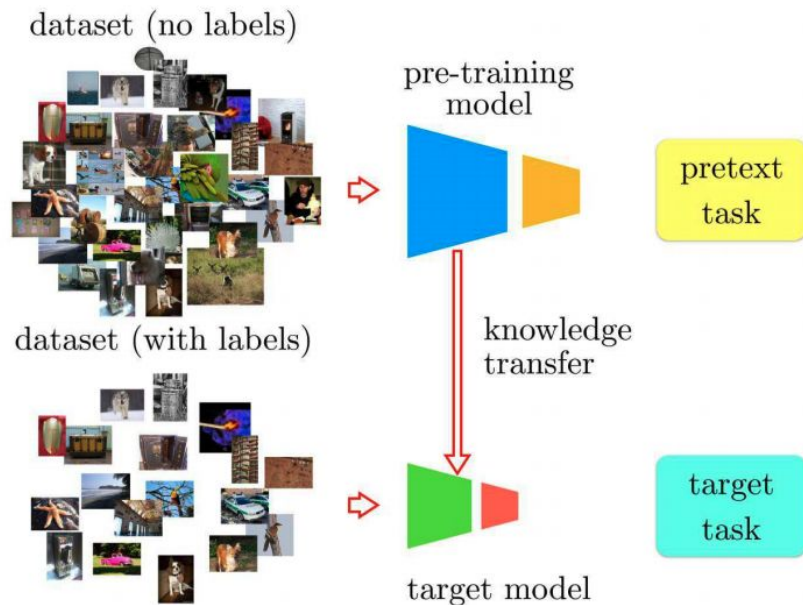
# Unsupervised Representation Learning By Predicting Image Rotations

**Authors:** Spyros Gidaris, Praveer Singh, Nikos Komodakis

**Presented by:** Akshay Singh Rana, Harmanpreet Singh

# Self Supervised Learning

- ConvNets are known to learn high-level semantic image features that usually require massive amounts of manually labeled data
- Building large labeled corpus is both expensive and infeasible to scale
- Define a pretext task for the ConvNet model to learn the image representations using self supervised learning



Reference: <https://arxiv.org/pdf/1805.00385.pdf>

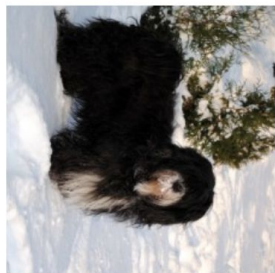
# Introduction

**Pretext task:** Classification task for the ConvNet to learn the geometric transformation applied to images. Learn image representations by training ConvNets to recognize the 2D image rotations.

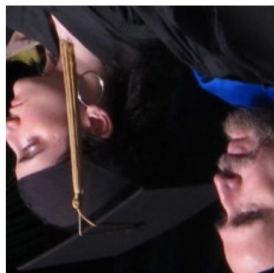
- Simple task, but actually provides a very powerful supervisory signal for semantic feature learning
- Force the ConvNet to learn semantic image features that can be useful for other vision tasks
- For model to recognize the rotation transformation, it will require to understand the concept of the objects depicted in the image



90° rotation



270° rotation



180° rotation



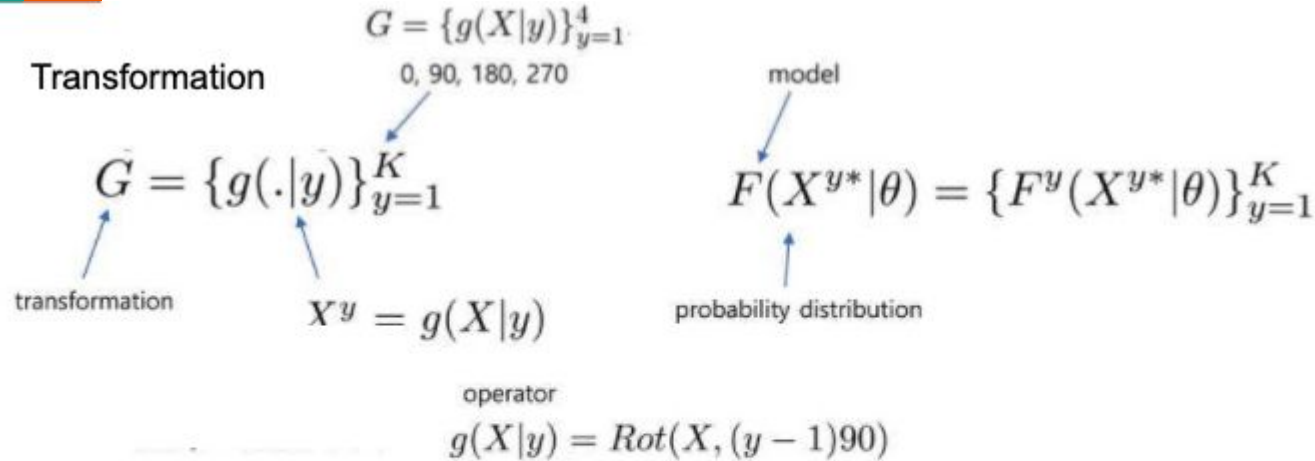
0° rotation



270° rotation

# Methodology

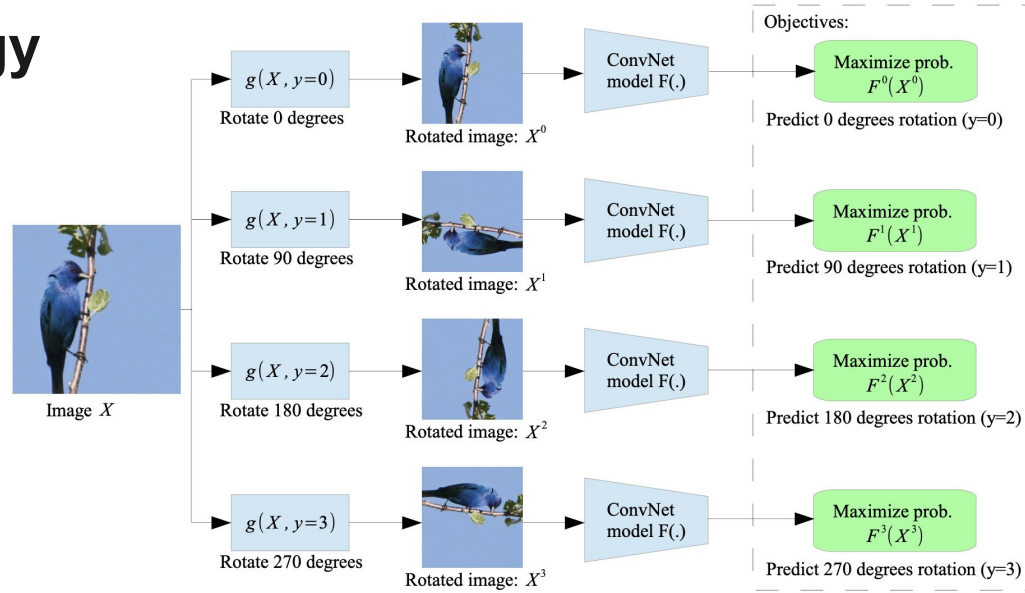
- Transformation



- Loss Function

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \text{loss}(X_i, \theta) \quad \text{loss}(X_i, \theta) = -\frac{1}{K} \sum_{y=1}^K \log(F^y(g(X_i|y)|\theta))$$

# Methodology



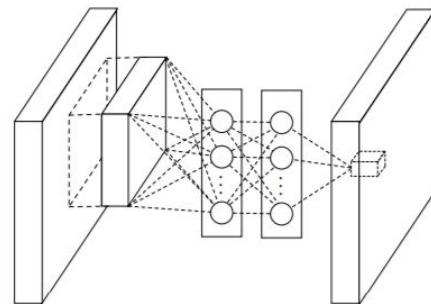
- ConvNet model  $F(\cdot)$  is trained to recognize the rotation (0, 90, 180, 270) that is applied to the image that it gets as input
- $F^y(X^{y^*})$  is the probability of rotation transformation  $y$  predicted by model  $F(\cdot)$  when it gets as input an image that has been transformed by the rotation transformation  $y^*$



# CIFAR Experiments

## RotNet model implementation details

- RotNet models implemented with Network-In-Network (NIN) architectures
- Three RotNet models which have 3, 4, and 5 convolutional blocks were trained
- Each block in the NIN architectures that implement RotNet models have 3 conv. layers
- Used SGD with batch size 128, momentum 0.9, weight decay  $5e-4$  and learning rate of 0.1
- Model trained for 100 epochs, learning rate dropped by factor of 5 after 30, 60 and 80 epochs
- After initial experiments, discovered significant improvement when during training the network was fed by all the four rotated copies of an image simultaneously, instead of each time randomly sampling a single rotation transformation. Therefore, at each training batch the network sees 4 times more images than the batch size



(b) MLPconv layer

Reference: <https://arxiv.org/pdf/1312.4400.pdf>


# Evaluation of learned feature hierarchies

Model	ConvB1	ConvB2	ConvB3	ConvB4	ConvB5
RotNet with 3 conv. blocks	85.45	88.26	62.09	-	-
RotNet with 4 conv. blocks	85.07	89.06	86.21	61.73	-
RotNet with 5 conv. blocks	85.04	<b>89.76</b>	86.82	74.50	50.37

*Explore how the quality of the learned features depends on their depth in the RotNet model. Table shows the classification accuracy on CIFAR-10 using the unsupervised learned features when trained a non-linear object classifier on top of them.*



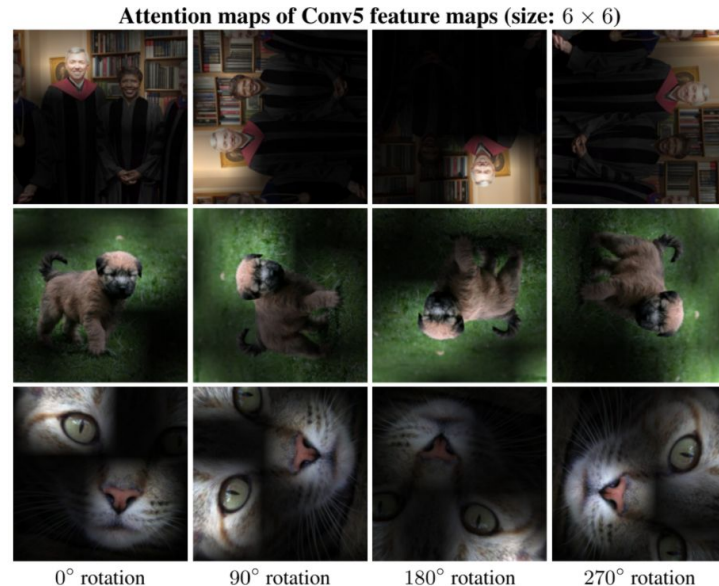
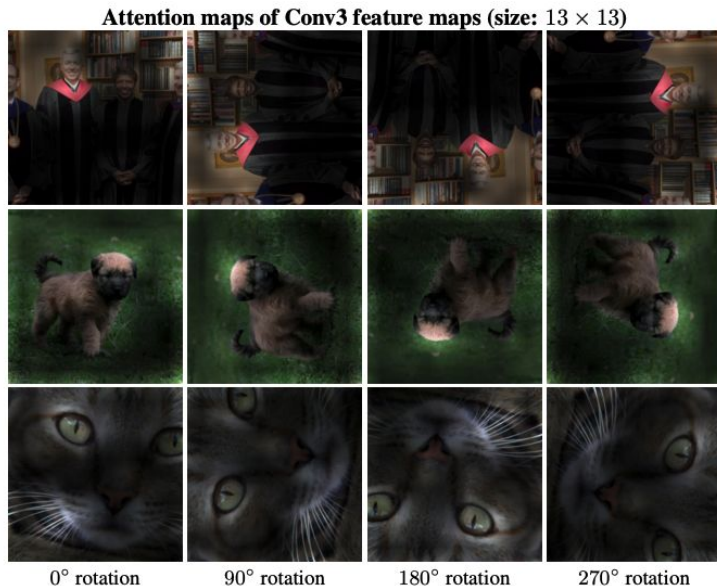
# Evaluation of the quality of learned features w.r.t. rotations



# Rotations	Rotations	CIFAR-10 Classification Accuracy
4	$0^\circ, 90^\circ, 180^\circ, 270^\circ$	<b>89.06</b>
8	$0^\circ, 45^\circ, 90^\circ, 135^\circ, 180^\circ, 225^\circ, 270^\circ, 315^\circ$	88.51
2	$0^\circ, 180^\circ$	87.46
2	$90^\circ, 270^\circ$	85.52

*For all the entries a non-linear classifier was trained with 3 fully connected layers on top of the feature maps generated by the 2nd conv. block of a RotNet model with 4 conv. blocks in total.*

# Visualization of Attention Maps



*Attention maps of the Conv3 and Conv5 feature maps generated by an AlexNet model trained on the self-supervised task of recognizing image rotations*

# Q & A

M

Maxime Daigle 1:54 PM

"In our preliminary experiments we found that we get significant improvement when during training we train the network by feeding it all the four rotated copies of an image simultaneously instead of each time randomly sampling a single rotation transformation. Therefore, at each training batch the network sees 4 times more images than the batch size."

They compare feeding the 4 rotations of each image in the same batch with feeding only one rotation of each image per epoch. How would it affect the performance if the 4 rotations are fed per epoch, but not necessarily simultaneously in the same batch? i.e. what if the dataset augmented by 4 rotations is shuffled? Would training on 1 epoch of the shuffled augmented dataset give a similar performance to training on 4 epochs of the dataset with one random rotation per image? Any intuition on why simultaneously feeding the transformations of an image would be better than shuffling them.

Why pass all 4 rotated images in same mini-batch?

Why only 4 rotations angles is 4 better than 8/2?

Should we drop 0 degree image and evaluate the #rotations to include?



Darshan Patil 1:22 AM

1. They mention that they feed in all 4 rotations of an image because that leads to better performance. Is that just a function of a larger batch size or does having all 4 rotations of an image somehow provide some kind of evening out effect?
2. When they compare 8 rotations vs 4 rotations, they say that for 8 rotations, they take care to crop the image to not include the empty image areas. Could this cropping be a reason for worse performance?



Maksym Perepichka 2:38 PM

The authors attribute the slightly worse performance of using 8 rotations compared to 4 to two possible factors: lack of distinguishability between fine rotations and visual artifacts present in images rotated by degrees that aren't multiples of 90. Perhaps naively, the following thought comes to mind: Couldn't the second part of this hypothesis be tested by training a system that only considers rotations that are NOT multiples of 90 (with no 0-degree rotation present) and comparing between 8 and 4 rotations?



Shanel Gauthier 2:11 PM

In Table 2, it is shown that they obtain better object recognition performance with 4 discrete rotations rather than 8. I was a bit surprised by this result. In fact, they said that adding 4 rotations can lead to visual artifacts. Can you elaborate more about this? (edited)



Erfan Salehi 11:57 AM

In Table 2, we see that the highest performance is gained when 4 rotation configurations are used to train the network. It is counterintuitive that including 4 more rotation configurations would make the neural net perform worse. They mention two likely causes for this.

1. The extra rotations could introduced may lead to visual artifacts on rotated images in a way that is detrimental to solving downstream tasks. Could you elaborate more on this and how this might happen?
2. The four extra rotation classes are not distinguishable enough to help with better representations for solving the downstream task. Could you elaborate more on this as well?

Is it possible that there might be other causes for this dropping of performance?



# Selected Answers



- **Why only 4 rotations angles / Why 4 rotations better than 8 or 2?**
  - Having just 2 rotations offers too few classes for recognition, that is providing less supervisory information.
  - The object recognition performance in 8 rotations is comparable to 4 class with the gap of just 0.5%.
- **Should we drop 0 degree image and evaluate the #rotations to include?**
  - Typically test images are present in landscape or portrait mode with angle of rotation to be equal to 0
- **Why passing all 4 rotated images in same mini-batch improves performance?**
  - During back-prop, we are interested to update weights together for seed image

# Q & A



**Pierre-André Brousseau** 11:36 AM

The authors present their attention map results in Figure 3 on images which feature heads. When an image has a head in it, it obviously offers very strong clues as to the orientation. Appendix A shows this in an interesting way as they seem to have built a head detector indirectly. Was there reason to believe that this task (learning rotations) would lead to learned features that cover a wide range of semantic definitions?



**milad aghajohari** 4:07 AM

**Q1:** I am wondering how non-trivial this task actually is. Looking at attention maps it seems to me that the model guess the rotations of persons by looking at their faces. Can we possibly say by just learning how to tell the rotation from a few short-cut objects like eyes the model is not forced to understand the semantics of the image thoroughly?

Table 8: **Per class PASCAL VOC 2007 detection performance.** As usual, we report the average precision metric. The results of the supervised model (i.e., ImageNet labels entry) come from Doersch et al. (2015).

Classes	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
ImageNet labels	64.0	<b>69.6</b>	<b>53.2</b>	<b>44.4</b>	<b>24.9</b>	<b>65.7</b>	<b>69.6</b>	<b>69.2</b>	28.9	<b>63.6</b>	<b>62.8</b>	<b>63.9</b>	<b>73.3</b>	64.6	55.8	<b>25.7</b>	<b>50.5</b>	55.4	69.3	<b>56.4</b>
(Ours) RotNet	<b>65.5</b>	65.3	43.8	39.8	20.2	65.4	69.2	63.9	<b>30.2</b>	56.3	62.3	56.8	71.6	<b>67.2</b>	<b>56.3</b>	22.7	45.6	<b>59.5</b>	<b>71.6</b>	55.3

Table 9: **Per class CIFAR-10 classification accuracy.**

Classes	aero	car	bird	cat	deer	dog	frog	horse	ship	truck
Supervised	<b>93.7</b>	<b>96.3</b>	<b>89.4</b>	82.4	<b>93.6</b>	<b>89.7</b>	<b>95.0</b>	<b>94.3</b>	<b>95.7</b>	<b>95.2</b>
(Ours) RotNet	91.7	95.8	87.1	<b>83.5</b>	91.5	85.3	94.2	91.9	<b>95.7</b>	94.2





**Tianyu Zhang** 3:27 PM

My question to submit for this paper:

How do they avoid overfitting during training to predict the rotation angle? For example, in this image, we want the model to learn what a book looks like. But actually, the model more likely (from my guess) infers from the location of the desk. (Thus, the attention may be on the desk not the book.) After all, as a human, if there are no desks in this image, I can hardly tell which rotation angle is right. I also need to refer the desk to predict. (edited)



**Ian Porada** 11:25 PM

I'm convinced by the empirical results in the paper that image rotations do not result in "low-level visual artifacts", yet intuitively I feel like there could be simple patterns, e.g. the relative position of blue pixels/the sky in landscape photos, that would be easy to detect without a deep semantic understanding. My question is then, do you think the conclusions of the paper would hold on a dataset with more homogeneous image classes such as landscapes? (edited)



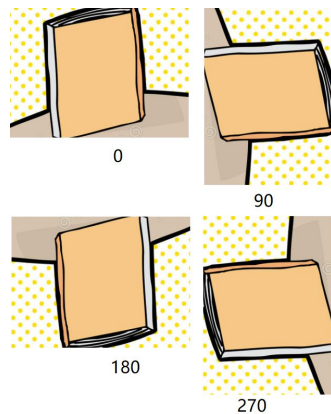
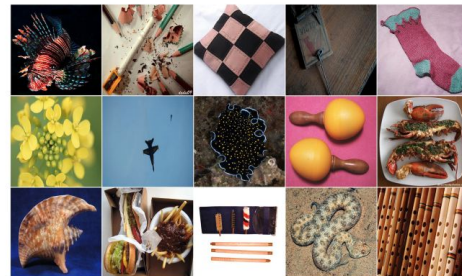
**Guillaume Lagrange** 3:13 PM

While the pretext task is intuitive and effective, the authors don't really mention difficult objects for which the direction is unclear or ambiguous (e.g. in Figure 2, the rotated image  $X^3$  could plausibly be associated to a  $0^\circ$  rotation). I understand that this doesn't apply to all objects (especially the ones in CIFAR-10), but do you think that the method wouldn't be as effective when using a more difficult dataset comprised of objects that have ambiguous directions (perhaps symmetrical objects), i.e. a lot more label noise? And perhaps as a follow-up, how do you suggest dealing with such noise if you think it could improve the method?

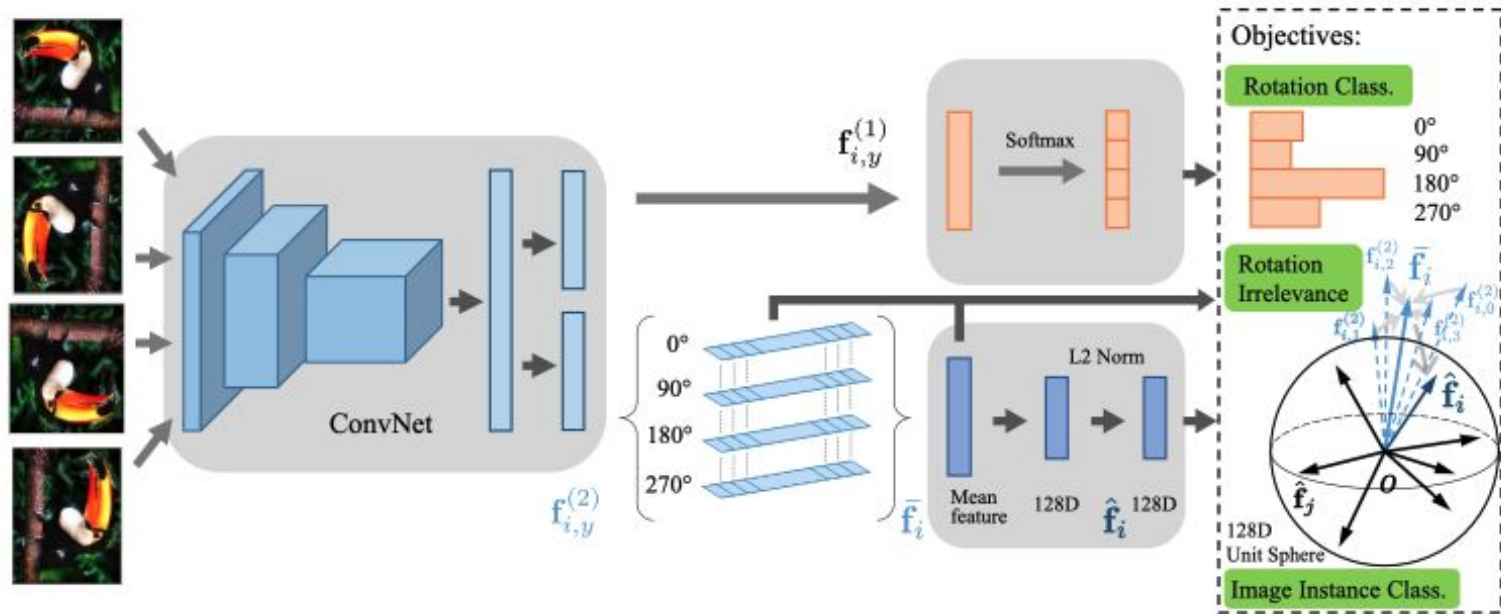


**Nitarshan** 9:12 AM

1. It's clear that some rotations may seem as normal (to a human) as the original orientation was. For example, the bird in Figure 2 is plausibly in a 0 degree orientation when actually rotated 270 degrees. I'd assume that at some point these plausible rotations would confound the learning process. How often is this likely to happen on something like CIFAR-10 or ImageNet? Would it make sense to prune away these image/rotation combinations that "incorrectly" place high probability scores on 0 degrees as training progresses?
2. The learned features are probably not perfectly invariant to the rotations applied. Would it make sense to average/concatenate over the rotated/flipped representations to build a better representation?



# Rotation Feature Decoupling



# Rotation Feature Decoupling - Objective

Rotation Classification

$$\mathcal{L}_c = \frac{1}{NK} \sum_{i=1}^N \sum_{y=1}^K w_{i,y} l(F_c(\mathbf{f}_{i,y}^{(1)}; \boldsymbol{\theta}_c), y),$$

Rotation Irrelevance

$$\mathcal{L}_r = \frac{1}{NK} \sum_{i=1}^N \sum_{y=1}^K d(\mathbf{f}_{i,y}^{(2)}, \bar{\mathbf{f}}_i).$$

features of all rotations to bring to a common mean and minimize it.

Image Instance Classification

$$\mathcal{L}_n = - \sum_{i=1}^N \log P(i | \hat{\mathbf{f}}_i).$$

mean features of all different images are maximized

$$\min_{\boldsymbol{\theta}} \frac{1}{NK} \sum_{i=1}^N \sum_{y=1}^K l(F(X_{i,y}; \boldsymbol{\theta}), y)$$

$$w_{i,y} = \begin{cases} 1 & y = 1 \\ 1 - \tilde{F}(X_{i,y})^\gamma & \text{otherwise.} \end{cases}$$

Probability of image being positive by a classifier trained on PU learning.


$$\min_{\boldsymbol{\theta}} \frac{1}{NK} \sum_{i=1}^N \sum_{y=1}^K w_{i,y} l(F(X_{i,y}; \boldsymbol{\theta}), y)$$

$$P(i | \hat{\mathbf{f}}) = \frac{\exp(\hat{\mathbf{f}}_i^\top \hat{\mathbf{f}} / \tau)}{\sum_{j=1}^N \exp(\hat{\mathbf{f}}_j^\top \hat{\mathbf{f}} / \tau)}$$

$$\min_{\boldsymbol{\theta}_f, \boldsymbol{\theta}_c} \lambda_c \mathcal{L}_c + \lambda_r \mathcal{L}_r + \lambda_n \mathcal{L}_n.$$



# RotNet - Results



Method	Accuracy	
Supervised NIN	92.80	←
Random Init. + conv	72.50	
(Ours) RotNet + non-linear	89.06	
(Ours) RotNet + conv	<b>91.16</b>	←
(Ours) RotNet + non-linear (fine-tuned)	91.73	
(Ours) RotNet + conv (fine-tuned)	92.17	←
Roto-Scat + SVM Oyallon & Mallat (2015)	82.3	
ExemplarCNN Dosovitskiy et al. (2014)	84.3	
DCGAN Radford et al. (2015)	82.8	
Scattering Oyallon et al. (2017)	84.7	

*Evaluation of unsupervised feature learning methods on CIFAR-10.*



**Max Schwarzer** 7:27 PM

Why should we expect performance for the fine-tuned variant of RotNet (table 3) to be worse than the supervised counterpart, given that they have identical architectures? Assuming the fine-tuning phase was trained to convergence, doesn't this imply that the features learned by RotNet actually *harmed* generalization in a way that couldn't be undone during fine-tuning?



**Abderrahim Fathan** 1:27 PM

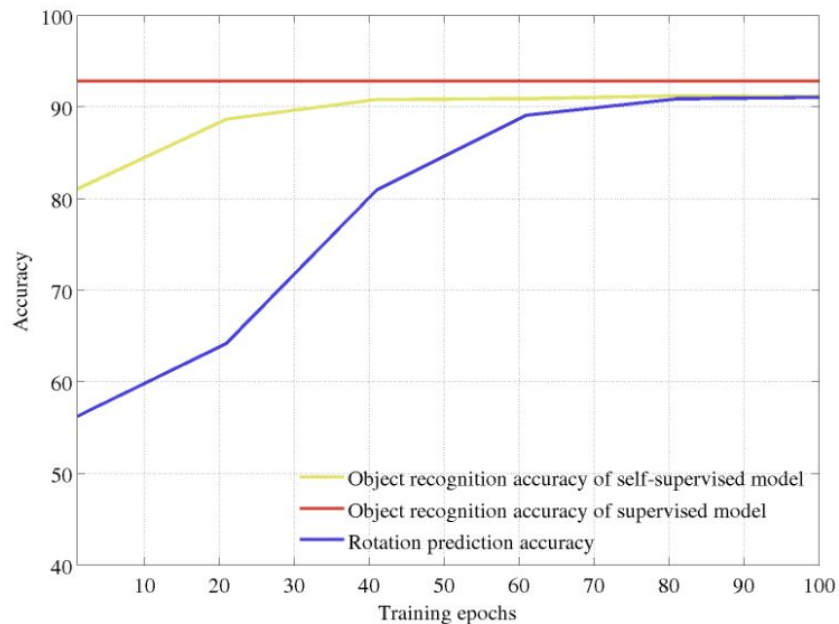
In Figure 5 (b) why the semi-supervised algorithm doesn't scale well with the number of training samples, to (why not) outperform the supervised algo. Is that a symptom that the transfer learning was not good and that the learned features are been gorgotten through time to learn new ones specific to the task. I find this especially contradictory with the fact that they claim « fine-tuning the unsupervised learned features further improves the classification performance. ». Could you please elaborate more on this?

Method	Accuracy
Supervised NIN	92.80
(Ours) RotNet + conv (fine-tuned)	92.17

	Top-1
Full sup.	59.4
Random	42.6
Scattering	49.2
BiGAN, A	51.4
RotNet, A	49.5
DeepCluster A	52.5

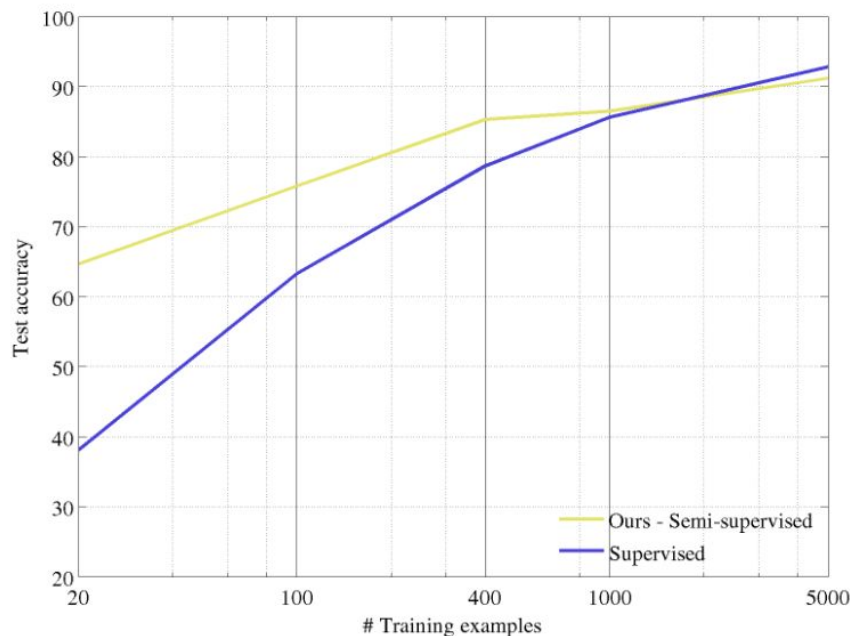
Reference: A CRITICAL ANALYSIS OF  
SELF-SUPERVISION

# Results



*Plot with the rotation prediction accuracy and object recognition accuracy as a function of the training epochs used for solving the rotation prediction task*

## Results: Semi-Supervised setting



*Plot with the rotation prediction accuracy and object recognition accuracy as a function of the training epochs used for solving the rotation prediction task*



# Imagenet Experiments

# Implementation Details

---

- RotNet model with an AlexNet architecture with no local response normalization units, dropout units, or groups in the convolutional layers
- RotNet model used batch normalization units after each linear layer (either convolutional or fully connected)
- Used SGD with batch size 192, momentum 0.9, weight decay  $5e-4$  and learning rate of 0.01
- Learning rate decreased by a factor of 10 after epochs 10, and 20 epochs. Model trained for 30 epochs
- As in the CIFAR experiments, during training the RotNet model was fed with all four rotated copies of an image simultaneously (in the same mini-batch)

# Results

Method	Conv4	Conv5
ImageNet labels from (Bojanowski & Joulin, 2017)	59.7	59.7
Random from (Noroozi & Favaro, 2016)	27.1	12.0
Tracking Wang & Gupta (2015)	38.8	29.8
Context (Doersch et al., 2015)	45.6	30.4
Colorization (Zhang et al., 2016a)	40.7	35.2
Jigsaw Puzzles (Noroozi & Favaro, 2016)	45.3	34.6
BIGAN (Donahue et al., 2016)	41.9	32.2
NAT (Bojanowski & Joulin, 2017)	-	36.0
(Ours) RotNet	50.0	43.8

*with non-linear layers..*

# Results

Method	Conv1	Conv2	Conv3	Conv4	Conv5
ImageNet labels	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Random rescaled Krähenbühl et al. (2015)	17.5	23.0	24.5	23.2	20.6
Context (Doersch et al., 2015)	16.2	23.3	30.2	31.7	29.6
Context Encoders (Pathak et al., 2016b)	14.1	20.7	21.0	19.8	15.5
Colorization (Zhang et al., 2016a)	12.5	24.5	30.4	31.5	30.3
Jigsaw Puzzles (Noroozi & Favaro, 2016)	18.2	28.8	34.0	33.9	27.1
BIGAN (Donahue et al., 2016)	17.7	24.5	31.0	29.9	28.0
Split-Brain (Zhang et al., 2016b)	17.7	29.3	35.4	35.2	32.8
Counting (Noroozi et al., 2017)	18.0	30.6	34.3	32.5	25.7
(Ours) RotNet	<b>18.8</b>	<b>31.7</b>	<b>38.7</b>	<b>38.2</b>	<b>36.5</b>

*with linear layers...*



# Results: 205-way Places

Method	Conv1	Conv2	Conv3	Conv4	Conv5
Places labels Zhou et al. (2014)	22.1	35.1	40.2	43.3	44.6
ImageNet labels	22.7	34.8	38.4	39.4	38.7
Random	15.7	20.3	19.8	19.1	17.5
Random rescaled Krähenbühl et al. (2015)	21.4	26.2	27.1	26.1	24.0
Context (Doersch et al., 2015)	19.7	26.7	31.9	32.7	30.9
Context Encoders (Pathak et al., 2016b)	18.2	23.2	23.4	21.9	18.4
Colorization (Zhang et al., 2016a)	16.0	25.7	29.6	30.3	29.7
Jigsaw Puzzles (Noroozi & Favaro, 2016)	<u>23.0</u>	<u>31.9</u>	35.0	34.2	29.3
BIGAN (Donahue et al., 2016)	22.0	28.7	31.8	31.3	29.7
Split-Brain (Zhang et al., 2016b)	21.3	30.7	34.0	34.1	<u>32.5</u>
Counting (Noroozi et al., 2017)	<b>23.3</b>	<b>33.9</b>	<b>36.3</b>	<b>34.7</b>	29.6
(Ours) RotNet	21.5	31.0	<u>35.1</u>	<u>34.6</u>	<b>33.7</b>

*All weights are frozen and feature maps are spatially resized (with adaptive max pooling) so as to have around 9000 elements.*

# Results: PASCAL VOC

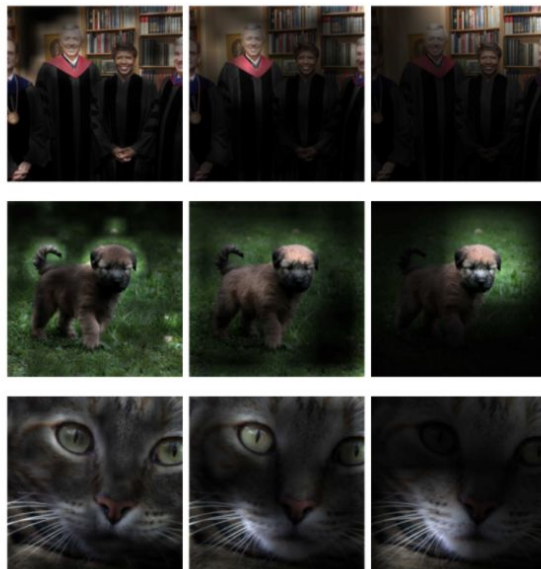
	Classification (%mAP)		Detection (%mAP)	Segmentation (%mIoU)
Trained layers	fc6-8	all	all	all
ImageNet labels	78.9	79.9	56.8	48.0
Random		53.3	43.4	19.8
Random rescaled Krähenbühl et al. (2015)	39.2	56.6	45.6	32.6
Egomotion (Agrawal et al., 2015)	31.0	54.2	43.9	
Context Encoders (Pathak et al., 2016b)	34.6	56.5	44.5	29.7
Tracking (Wang & Gupta, 2015)	55.6	63.1	47.4	
Context (Doersch et al., 2015)	55.1	65.3	51.1	
Colorization (Zhang et al., 2016a)	61.5	65.6	46.9	35.6
BIGAN (Donahue et al., 2016)	52.3	60.1	46.9	34.9
Jigsaw Puzzles (Noroozi & Favaro, 2016)	-	67.6	53.2	37.6
NAT (Bojanowski & Joulin, 2017)	56.7	65.3	49.4	
Split-Brain (Zhang et al., 2016b)	63.0	67.1	46.7	36.0
ColorProxy (Larsson et al., 2017)		65.9		38.4
Counting (Noroozi et al., 2017)	-	67.7	51.4	36.6
(Ours) RotNet	<b>70.87</b>	<b>72.97</b>	<b>54.4</b>	<b>39.1</b>

*For classification, we either fix the features before conv5 (column fc6-8) or we fine-tune the whole model (column all). In this case the learnt features are evaluated w.r.t. their generalization on classes that were “unseen” during the unsupervised training phase*

# Attention Maps

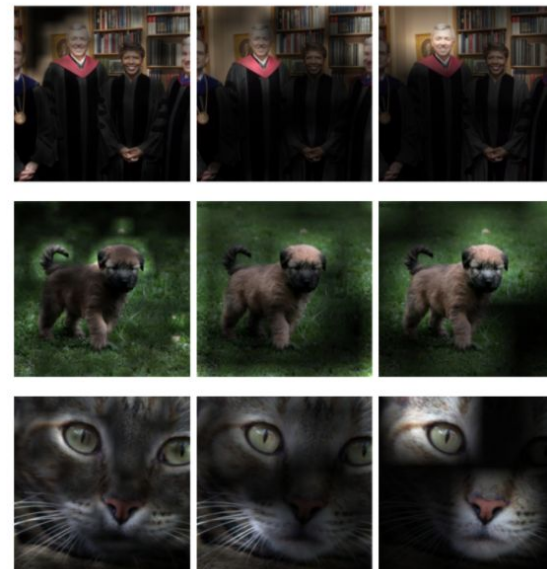


Input images on the models



Conv1  $27 \times 27$  Conv3  $13 \times 13$  Conv5  $6 \times 6$

(a) Attention maps of supervised model

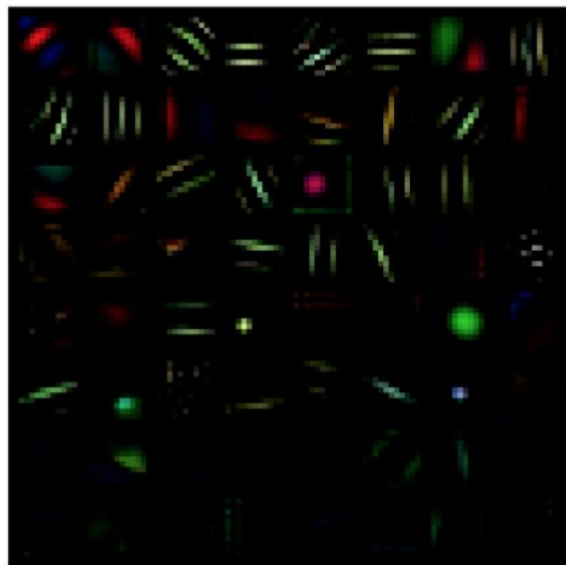


Conv1  $27 \times 27$  Conv3  $13 \times 13$  Conv5  $6 \times 6$

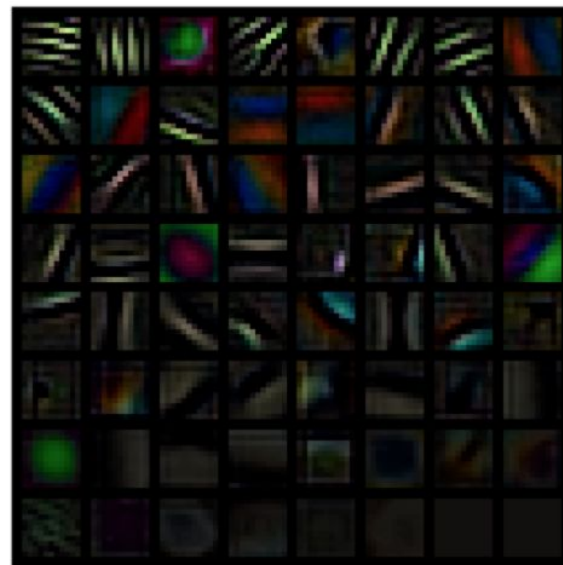
(b) Attention maps of our self-supervised model

*Attention maps generated by an AlexNet model.*

# Feature Maps



(a) Supervised



(b) Self-supervised to recognize rotations

*First layer filters learned by a AlexNet model*

# Q & A



**Ethan Caballero** 7:54 AM

In figure 4, why do the self-supervised vs supervised filters look different in the ways that they do?



**Francois Mercier** 11:51 PM

replied to a thread: [In figure 4, why do the self-supervised vs supervised filters look different in the ways that they do?](#)

To follow up this question, do you think the reason for richer filters in SSL setting is because the pretext class helps to learn features which are more generalizable than the supervised learning setting?



**Laleh Touraj** 11:00 PM

My question gets back to figure 4. How does the difference between filter learned in (a) and (b) could be justified. Plus, the author indicated [filters learned by SSL approach have more variety than supervised task](#) does learning more filter necessarily means the method do better in terms of learning representation ?

# Q & A

---



**Nicolas Trudel-Mallet** 2:03 PM

Do you think generalization performance could be improved by combining this rotation-prediction task with one or more others (e.g. colorization, spatial context...)? Are there any other pretext tasks that would be particularly complementary to this one? (edited)



**Marc Andre Ruel** 2:45 PM

We already saw that other data augmentations (color jitters, cropping, etc) can help produce better results depending on the task. Do you think that adding such augmentations in this case would help learn better representations (help performance) even though we are only trying to predict the image orientation?




**Makesh Narasimhan** 7:37 AM

Would this pretext task work well even for grayscale images? (Any intuition on how important colored images are for learning good features?) I think that while grayscale images would work relatively fine for images with well-defined objects like vehicles or persons, color might play an important role in images of nature (water bodies, etc.). It would be good to know your thoughts on this.

# Q & A

---

 **Eeshan Gunesh Dhekane** 1:21 AM

Based on Tables 4 to 7, it can be said that predicting the image rotation is a better pretext task as compared to those introduced in previous approaches (like the context prediction or solving jigsaw puzzles). All these pretext tasks are designed with the aim that they force the network to learn semantic representations of the data. So, I would like to know your opinion on this general question-- Why is it the case that this particular pretext task is better than the previous ones? Is there a particular reason according to you that this particular task is indeed better than the previous ones?