Self-Supervised learning using Rotation prediction in Object and Scene Centric datasets

Joseph Daniel Selvaraaj

Abstract

As most of the data available are unlabeled, there has been an increased interest in research in unsupervised learning, specifically self-supervised learning. Many different augmentation techniques have been suggested in the last decade, including unsupervised representation learning by predicting rotations. It has been shown that such methods were on par with the state-of-the-art supervised models trained with labels. However, in the past, the augmentation methods have been tested on object-centric datasets such as ImageNet. Moreover, they used deep models such as AlexNet, which has 61M parameters. The main goal of this project is to analyze the self-supervised learning method (by rotation prediction) on one object centric dataset (Labelled Faces in the wild) and two scenes centric datasets (COCO and Cityscape) with three different architectures – a custom six-layer deep network, EfficientNet b0, and MobileNet v2 - with the parameter size of custom six-layer net being around 100k, MobileNet v2 being around 3.4M, EfficientNet b0 being around 5.3M. The saliency maps of the models trained via unsupervised learning are compared with the saliency maps of the respective randomly initialized model. The analysis suggests that these relatively tiny models learn low-level features such as edge detection to solve the rotation problem. It also suggests that rotation as an augmentation technique for self-supervised learning might not work well for all datasets.

1. Introduction

Many of the current State of Art models were trained from large amounts of supervision from labeled data. However, accruing labels is expensive in many applications. In the last few years, researchers have found that until the last few layers (that is, till the embedding layer), the networks learn the low-level features from the images, which are usually data (image) agnostic.

It has been hypothesized and empirically shown that for learning most of these low-level feature labeled data is not required. If a network could be trained to learn most of the low-level features without labels, then such a model, when trained with labels for downstream tasks by freezing all layers except the last few, would have a relatively low number of parameters and hence would require less number of labeled images to train.

Many methods have been proposed to make a network learn this latent information (the application agnostic features) without labels. One such method is to train a network in a supervised manner for a different problem for which labels could be generated within the algorithm. This type of unsupervised learning is called self-supervised learning. The intuition behind this method is that it is tough for the network to directly solve the new made-up problem without first learning all the low to high-level features.

One such method in self-supervised learning that has been previously successful in making a network learn representations is to train a network to predict the degree of rotation of a given image from a small subset [1]. Prior work has been done on unsupervised learning by training the network to predict rotation on only object-centric datasets with older models.

This project trained a few new models with much less parameter size with these unsupervised learning methods on scene-centric datasets. The main problem is to learn the latent representation of images without labels for any dataset.

We evaluate the model based on the annotated labels for COCO's typical object detection task. First, compute the saliency maps of the network trained on the self-supervised tasks. Then, a threshold value normalizes the saliency maps. If there is more than one saliency map for the same image, we can either compute the average

saliency map or max of all the saliency maps. Then compute the amount of activity inside the bounding boxes from COCO. The details are explained later.

2. Background and Related Work

2.1 Unsupervised learning

The fundamental idea of most unsupervised learning methods is to train a network to learn a mapping from the input dataset to a lower dimension representation space without using any annotated labels. This process can be thought of as training a compressor (a network that maps inputs from a very high dimensional space, input space, to a much lower dimension representation space). There are many types of unsupervised learning, such as autoencoders (which have a bottleneck architecture that forces the network to learn the representations), GANs (the discriminator learns the representation to distinguish fake generated inputs from real inputs), clustering (such as k-means clustering) and Self-Supervised learning.

2.2 Self-supervised learning

Self-supervised learning is best defined as a paradigm of unsupervised learning in which a pretext task is used as a surrogate supervision signal to learn many low- and high-level features in a given dataset (Gidaris et al., 2018).

There is a wide variety of pretext tasks. Some of the heuristic based Self-supervised methods include hallucinating a plausible color of a grayscale image (Zhang et al., 2016), combining multiple pretext tasks into one (Doersch et al., 2015), solving jigsaw puzzles (Noroozi et al., 2016), predicting rotation (Gidaris et al., 2018). Other self-supervised learning approaches include contrastive learning (SimCLR, MOCO), Unsupervised Learning by Predicting Noise (Piotr et al., 2017).

The intuition behind any pretext task is that since the tasks are trained with deep CNNs, the networks first use their receptive field to recognize objects in the images, which is then used to do well in the pretext task. Note that we are only interested in the representations, which is the layer before the pretext task-specific layers.

2.2.1 Colorizing grayscale images

The pretext task in this method is to train the network with the grayscale version of the unlabeled dataset to predict the hue and chroma distribution per pixel.

2.2.2 Unsupervised learning by solving jigsaw puzzles

The pretext task is to divide the input image into a constant number of patches (let us say 6) and train the network to predict the position of a given patch in the original image.

2.2.3 Rotation prediction

The network is trained to predict the rotation (from a given subset such as {0,180}) given various rotated versions of an image.

2.2.4 SimCLR

SimCLR is a different approach from the previous pretext of heuristic-based self-supervised learning. In SimCLR, the network works directly on the latent space, unlike the previous methods, which created a surrogated task, and worked on the output space of those tasks. The methodology is as follows. A new dataset is constructed from the unlabeled dataset containing pairs of images. Each image is generated by different transformations to the original images. A pair is considered positive if both the images in it are transformations of a single image from the original dataset; negative otherwise. The network is then trained to attract positive pairs and repel negative pairs by designing the loss function to have a higher loss if negative pairs are mapped to similar representations or positive pairs are mapped to very different representations.

2.2.5 Noise Prediction

This method is similar to SimCLR as this method also works directly on the latent representation space. The network is trained to map each image from the unlabeled dataset to a random projection sampled from a spherical gaussian distribution [2]. The intuition behind this is that since the deep features are constrained, the network is forced to align the noises (random projections), which is shown to be a good representation for the initial dataset.

2.3 Descriptions of the Networks

Efficient net is a family of models which has achieved better accuracy than ResNet and DenseNet in ImageNet, with 8.4x smaller and 6.1x faster. It has around 5.3 million parameters [3]. Note that this model has residual connections.

MobileNet has around 3.4 million parameters. It is a widely used model [4]. Note that this model has residual connections.

Six-layer network is made of convolutions, Batchnorm, and ReLU layers. It has around 100 thousand parameters. Note that this model does not have residual connections.

2.4 Saliency Map

A saliency map is a standard method used for visualizing a neural network to get an idea of what the network is learning. Saliency maps are usually calculated for a single neuron. It is usually framed as the following

$$S = \nabla f(x)$$

Where x is the input image and f is the output of a neuron we are interested in.

S has the same dimension as the input image, and each value of S can be interrupted as "how much does the neuron output change if we were to change the pixel value at a specific location."

3. Approach

3.1 Problem Formulation

The goal of the network is to predict the geometric transformation of a given image.

Let the original dataset be $D = \{x_1, x_2, x_3 \ldots . x_n\}$

We then apply a geometric transformation g(x,θ) where x= image and θ= the angle by which the image has to be rotated.

A new dataset

$D' = \{g(x_1, 0), \ldots g(x_n, 0), g(x_1, 90),$

$\ldots g(x_n, 90), g(x_1, 180), \ldots g(x_n, 180), g(x_1, 270),$

$\ldots g(x_n, 270)\}$

$|D'| = 4n$

Task: Train a supervised deep ConvNet on D'

3.2 Implementation details

3.2.1 Preprocessing

The datasets were normalized with the ImageNet mean and standard deviation as per the typical convention. All the images were resized to 128 by 128 using bilinear interpolation.

3.2.2 Training Details

The network was trained using the cross entropy loss, RMS prop optimizer and exponential decay learning rate scheduler. All the networks for all datasets were trained with the same hyperparameters which are as follow

RMS prop:

Learning rate = 1e-4

Momentum = 0.5

Exponential decay learning rate scheduler:

Step size for exponential decay = 10

Regularization = 1

4. Experiments

4.1 Cityscape (Scene Centric)

A dataset consisting of complex scene-centric urban street images taken from 50 cities in Europe over three different seasons and various weather conditions. The images contain many objects, including flats, construction, nature, vehicle, the sky, and humans [6]. This project uses a mutually disjoint set of 2975 train images and 1525 test images.

4.2 COCO (Scene Centric)

A dataset consisting of a wide array of scenes containing any of the 80 object categories [5]. This project uses a subset of the COCO dataset containing images of dogs and cats. A mutually disjoint set of 8291 train images and 867 test images was used for this project.

This dataset also has bounding boxes around every labeled object in the images. This bounding box will be later used to quantify the saliency maps.

4.3 Labelled Faces in the Wild (Object Centric)

A dataset consisting of unconstrained object-centric faces images with a diverse collection of age, gender, race, pose, lighting, etc [7]. This project uses a mutually disjoint set of 7939 train images and 2647 test images.

4.4 Algorithm

For dataset in [Labelled faces in the wild, Cityscape]:

For each model:

- Prepare the labeled dataset by rotating every image by 0, 90, 180, and 270.
- Shuffle the dataset.
- Train the network on this new dataset.
- Compute saliency map of the rotation layer (detailed later).
- Compute saliency map of the rotation layer of a random (untrained) network for baseline.

For the COCO dataset:

For each model:

- Follow the same steps as above.
- Quantify and analyze the saliency maps.

4.5 Evaluation method

The goodness of the saliency map can be quantified for the COCO dataset since we have bounding boxes for this dataset.

The saliency maps are first denoised using a threshold. That is, after this process, there is an activation (greater 1) only if the activation value is greater than the threshold.

$$f(X,\lambda) = \begin{cases} 0 & if\ X_i < \lambda \\ 1 & otherwise \end{cases}$$

Where f is the denoising function X is the image, and λ is the threshold.

Let us call the saliency map after applying f as S_p, and let B be an image that has the same size as the input image and 1s in the coordinates inside the bounding box and 0 elsewhere. Then, the goodness of the saliency map is given by

$$Q(S,B) = \frac{e^T(S .* B)e}{e^T Be + e^T\ max\big((S-B),0\big)\ e}$$

Where $e = \vec{1}$ of size of one dimension of the image in our case, this would be 128. Since the images are 128 by 128.

$Q(S,B)$ can be conceptually thought of as calculating

$$\frac{S\ \cap B}{S\ \cup\ B}$$

Generalization of Saliency Map for more than one neuron

The following operation is performed to extend the definition of a saliency map for more than one neuron.

Let us say we want to calculate one saliency map for neurons $a_1, a_2, \ldots . a_n$

First, calculate the individual saliency maps using the original method (gradient w.r.t input) to get the saliency maps $s_1, s_2, \ldots . s_n$ Respectively.

The overall saliency map is given by $S = \max(s_1, s_2 \ldots s_n)$, where the max operation is applied elementwise across every saliency map. The intuition behind this is that since we are interested in finding the locations in the pictures which are used by any of the neurons, the S from the max operation would be one if at least that location influenced the activation of at least one neuron. This formula is used to generalize the saliency for both the rotation prediction layers and embedding layers.

Saliency Map

For COCO dataset the accuracy of the networks on rotation prediction are described below:

Accuracy	Efficient Net b0	Mobile Net v2	Six-Layer Net
COCO	54.44%	43.41%	57.59%
Labelled face in the wild	84.33%	60.36%	98.04%
Cityscape	99.75%	99.85%	99.35%

The average goodness of saliency maps across the test set for the rotation prediction layer is described below:

Network (Threshold)	Efficient net (8e-3)	Mobile Net (8e-3)	Six-layer Net (8e-4)
Baseline	0	0	0
COCO	28.68%	31.84%	12.81%

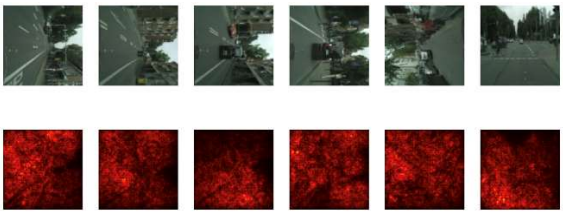
The average goodness of saliency maps across the test set for the embedding layer is described below:

Network (Threshold)	Efficient net (1e-4)	Mobile Net (1e-2)	Six-Layer Net (8e-4)
Baseline	0	0	0
COCO	11.61%	21.01%	6.59%

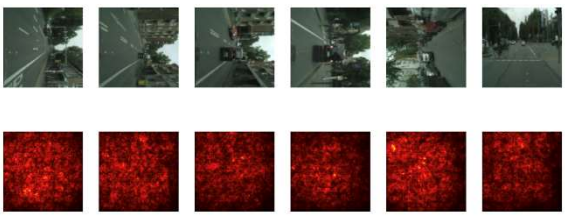
The saliency maps are visualized for the other two datasets.

Cityscape

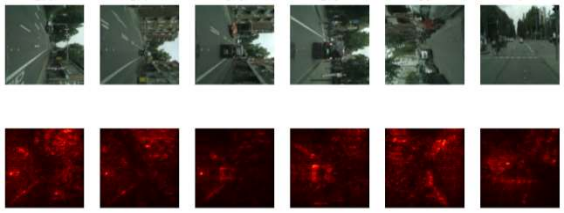
Mobile Net – Trained



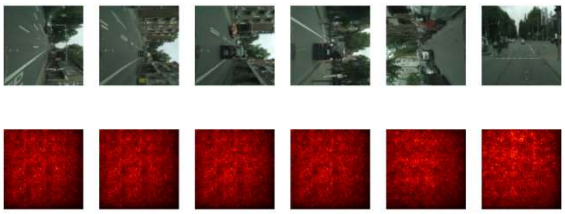
Mobile Net – Random



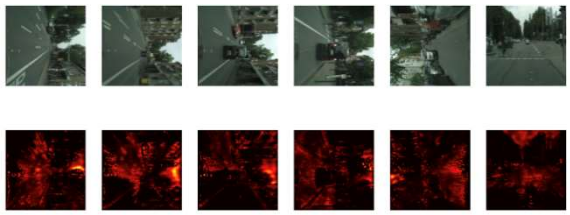
Efficient Net b0 – Trained



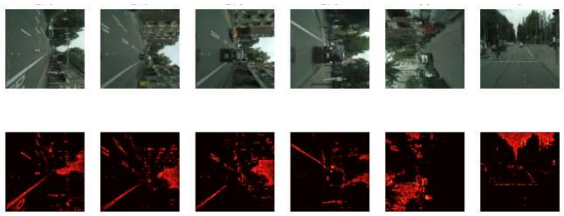
Efficient Net b0 – Random



Six-layer Net – Trained

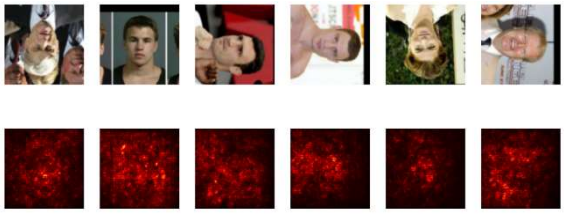


Six-layer Net – Random

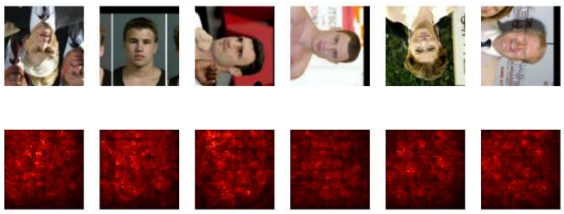


Labelled Faces in the Wild

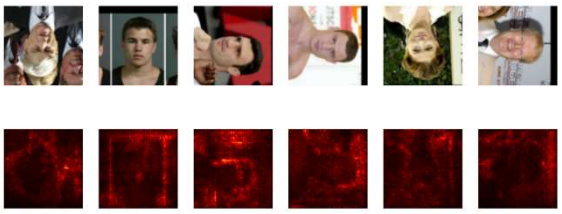
Mobile Net- Trained



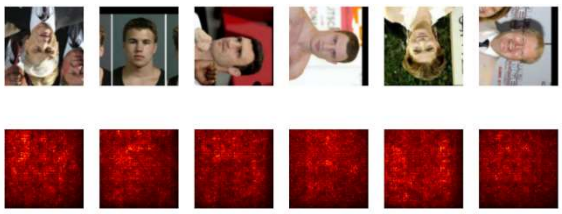
Mobile Net – Random



Efficient Net b0 - Trained

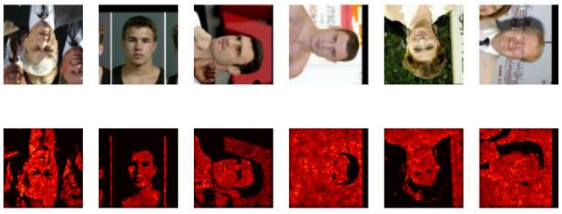


Efficient Net b0 – Random



Six-Layer Net– Trained

Six-Layer Net – Random



5. Conclusions

The results suggest that self-supervised learning by rotation prediction does learn good representations for some datasets such as Labelled faces in the wild and very poorly for Cityscape.

One possible reason could be that there are many straight edges in the Cityscape images. Maybe the network uses the angle of these straight edges to cheat in rotation prediction rather than isolating objects in the image

The Saliency maps of Cityscape also suggest that some networks tried to use the sky's brightness to predict rotation, which resulted in poor object recognition but very high rotation prediction accuracy.

Rotation prediction as self-supervised learning performs well when the images in a dataset

- Approximately rotationally symmetric with respect to the magnitude of the brightness values. (This is supported by poor localization in Cityscape)
- The background is rotationally symmetric but not the object themselves. In this case, the network is forced to isolate the object to predict image rotations and cannot use any other information. This hypothesis is supported by the excellent localization of the face in the saliency map of the Labelled Faces in the wild dataset.

In the future, the models could further be trained by freezing all layers except the last few on some downstream tasks based on annotated labels from the dataset and compare the model's accuracy to a random baseline trained by freezing the same layers.

6. References

[1] Spyros Gidaris, Praveer Singh, Nikos Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. arXiv:1803.07728v1 [cs.CV] 21 Mar 2018

[2] Piotr Bojanowski, Armand Joulin. Unsupervised Learning by Predicting Noise. arXiv:1704.05310v1 [stat.ML] 18 Apr 2017

[3] Mingxing Tan, Quoc V. Le . EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv:1905.11946v5 [cs.LG] 11 Sep 2020

[4] Mark Sandler Andrew Howard Menglong Zhu Andrey Zhmoginov Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381v4 [cs.CV] 21 Mar 2019

[5] Tsung-Yi Lin Michael Maire Serge Belongie Lubomir Bourdev Ross Girshick James Hays Pietro Perona Deva Ramanan C. Lawrence Zitnick Piotr Dollar. Microsoft COCO: Common Objects in Context. arXiv:1405.0312v3 [cs.CV] 21 Feb 2015

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos1, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. arXiv:1604.01685v2 [cs.CV] 7 Apr 2016

[7] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007.