

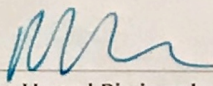


## APPROVAL SHEET

Title of Thesis: Boosting Self Supervised Learning via Knowledge Transfer

Name of Candidate: Ananthachari, Kavalkazhani Vinjimoor  
Master of Science, 2018

Thesis and Abstract Approved: \_\_\_\_\_

  
Dr. Hamed Pirsavash  
Assistant Professor  
Department of Computer Science  
and Electrical Engineering

Date Approved: 04/25/18

# ABSTRACT

**Title of Thesis:** Boosting Self-Supervised Learning via Knowledge Transfer

Ananthachari KV, M.S. Computer Science, February 2018

**Thesis directed by:** Dr. Hamed Pirsiavash, Assistant Professor  
Department of Computer Science and  
Electrical Engineering

In Self-supervised learning (SSL), an auxiliary task is designed to solve a particular problem, also called pretraining on a specific dataset without the need for human annotation. This process is an initial phase of transfer learning, where one learns a model on an auxiliary task and transfer this model to solve another task by fine-tuning on target dataset. In transfer learning, the inherent constraint is to use same model architecture for both pretraining and fine-tuning. This approach gives rise to issues in designing and comparing various models and auxiliary tasks. An example is, one cannot use different model architecture on auxiliary task and target domain task due to the limitations of fine-tuning and training settings. Since model architectures with varying task complexities are being used by researchers, this makes it hard to compare different approaches. The motive of this work is to design a framework that overcomes the above-mentioned limitations. If there is a way to transfer knowledge from a pre-trained model to a target model, then we should be able to use various architectures in both the phases.

Towards this goal, we designed a novel framework that separates the auxiliary training from target domain training by developing an effective transfer method based on clustering. We cluster the features computed from pretrained model to obtain pseudo-labels and learn a novel representation to predict the pseudo-labels. The intuition behind this approach is, in a good visual representational space, semantically

similar data points must be closer together than dissimilar data points. This metric should be learnt inherently by the network during the pre-training in order generate good features. This approach gives us flexibility in assessing incompatible models like hand-crafted features. This separation enables us to use different model architectures during auxiliary training and target domain training and also experiment with deeper models to learn better representations. We are also able to boost the performance by increasing the complexity of the auxiliary task and then transfer the knowledge from a deeper model to a shallower one. We conducted various experiments on various datasets to evaluate the performance of this method. This framework outperformed all the current state-of-the-art SSL methods on benchmarks datasets. Our method achieved 72.5% mAP on classification task and 57.2% mAP on object detection task of PASCAL VOC dataset.

*Keywords:* Image Classification, Convolutional Neural Networks, Object Detection, Self-supervised learning, Knowledge Transfer

# **Boosting Self-Supervised Learning via Knowledge Transfer**

by

Ananthachari Kavalkazhani Vinjimoor

Thesis submitted to the Faculty of the Graduate School  
of the University of Maryland in partial fulfillment  
of the requirements for the degree of  
M.S. Computer Science  
2018





*Dedicated to my friends and family who made the journey worthwhile and without  
them I would have finished my thesis an year ago.  
To my advisor who guided me to my destination*



## ACKNOWLEDGMENTS

I would like to thank my research adviser and mentor Dr. Hamed Pirsiavash for guiding me through my research, and for encouraging me to pursue and tackle interesting, challenging problems. He has been a great source of inspiration right from the beginning and the sole reason behind my decision to take up a career in this field. Having studied Electrical engineering during my undergrad, initially I was quite unsure about my career path and I took Introduction to Machine Learning and Computer Vision course taught by Hamed and I haven't looked back since. I would like to thank Hamed for being very patient with me and motivating me when part of the research failed. It has been an amazing 3 years of collaboration through coursework and research. I would like to thank him for giving me a chance to be a part of the paper which later became my thesis topic. I thank my collaborators Mehdi Noroozi and Paolo Favaro from University of Bern, Switzerland for their contributions and I learnt a lot throughout this project. Also, I would like to thank Dr. Tim Oates and Dr. Cynthia Matuszek for agreeing to be on my thesis evaluation committee.

I am fortunate to have had so many amazing friends throughout my graduate studies. I had a very close-knit group of people with whom I share beautiful memories and would cherish those moments forever. Without Arjun, Abhil and Nikita, I wouldn't have survived graduate school. They became my family away from home. I thank them for their support during my studies and they still continue to do so. I'd like to thank Ashwin, Vipin and Agniva for all the guidance and discussions. Pooja, Amrita and Bharat for all the support and also for making this journey more memorable. Also, my gratitude to my colleagues Rose, Rahul, Maryam, Erfan, Aniruddha and Akshay at the Computer Vision Lab whose cooperation and immense support

made my journey smooth and successful.

The success of my work would not have been possible without the support of my parents and my siblings, Ramya and Srivatsan, Rangarajan and Shwetha, Raghuram.

# TABLE OF CONTENTS

<b>DEDICATION</b>	<b>ii</b>
<b>ACKNOWLEDGMENTS</b>	<b>iii</b>
<b>LIST OF FIGURES</b>	<b>viii</b>
<b>Chapter 1 INTRODUCTION</b>	<b>1</b>
<b>Chapter 2 BACKGROUND AND RELATED WORK</b>	<b>6</b>
2.1 Convolutional Neural Network	6
2.1.1 Alexnet	7
2.1.2 VGG-16	8
2.2 Self-supervised Learning	9
2.3 Knowledge Distillation	11
<b>Chapter 3 METHODS</b>	<b>12</b>
3.1 Knowledge Transfer	12
3.1.1 Pre-training	13
3.1.2 K-Means	13
3.1.3 Pseudo labels	15

3.1.4	Cluster Classification . . . . .	15
3.2	Jigsaw++ . . . . .	15
<b>Chapter 4</b>	<b>DATASET . . . . .</b>	<b>18</b>
4.1	ImageNet . . . . .	18
4.2	PASCAL VOC . . . . .	19
4.3	PLACES . . . . .	20
<b>Chapter 5</b>	<b>EXPERIMENTS . . . . .</b>	<b>21</b>
5.1	Jigsaw++ trained with VGG16 . . . . .	22
5.2	Jigsaw++ trained with AlexNet . . . . .	22
5.3	Context Prediction . . . . .	22
5.4	Histogram of Oriented Gradients . . . . .	23
5.5	Random Pseudo-labels . . . . .	23
5.6	Knowledge Distillation . . . . .	23
5.7	Implementation Details . . . . .	24
<b>Chapter 6</b>	<b>ABLATION STUDIES . . . . .</b>	<b>25</b>
6.1	Impact of Number of Clusters . . . . .	25
6.2	Impact of Data Domain . . . . .	26
<b>Chapter 7</b>	<b>TRANSFER LEARNING EVALUATION . . . . .</b>	<b>28</b>
7.1	PASCAL VOC Fine-tuning . . . . .	28
7.2	Linear Classification . . . . .	31
7.3	Nonlinear Classification . . . . .	34

Chapter 8	VISUALIZATIONS . . . . .	35
Chapter 9	CONCLUSION . . . . .	38
	REFERENCES . . . . .	39

## LIST OF FIGURES

1.1	One of the challenges we address in this work is the need to use same architecture for pretraining and fine-tuning. Identical models and settings are used for source task and target task. . . . .	3
2.1	AlexNet architecture . . . . .	8
2.2	VGG16 architecture . . . . .	9
3.1	The steps involved in knowledge transfer of our framework. (a) pre-training on pretext task. (b) extract features using the pretrained model. (c) assign pseudo-labels by performing clustering. (d) train a new model from scratch on the dataset to predict the pseudo-labels .	14
3.2	A dataset example for Jigsaw++ pretext task where we divide an image into 9 tiles and randomly replace up to 2 tiles from a random image. .	17
8.1	Visualizations of conv1 weights, [a] randomly initialized AlexNet network, [b] HOG features, [c] Doersch et al. trained with color dropping, [d] Doersch et al. trained with color projection, [e] the task jigsaw++ trained on AlexNet, [f] the task jigsaw++ trained on VGG16. . . . .	36
8.2	Some cluster samples used to train jigsaw++ <sup>vcc</sup> . Each row shows the 11 closest images to their corresponding cluster center. . . . .	37

## Chapter 1

# INTRODUCTION

Most of the approaches to solve core computer vision problems employ supervised learning methodologies that involve human annotations. These approaches (1; 2; 3; 4) usually employ neural networks that achieve state-of-the-art performances on popular tasks like object classification, detection and segmentation, even exceeding human performance in few tasks. Convolutional neural networks have become the dominant approach for visual object recognition. With advancements in computer hardware and improvements in network architecture, it has become feasible to train on large amounts of data. The only limitation in this approach is the cost involved to annotate huge amounts of data. Collecting myriad of images from the internet is a cheap process but labelling these images is tedious and expensive in terms of time and money. Recently, to circumvent the problem of needing a huge amount of annotated data for training, an interesting approach called as Self-supervised learning was introduced. This approach has become very popular as it doesn't involve any human annotation. Self-supervised learning methods can learn visual features without manual labels by designing a so-called pre-text task or auxiliary task. Through transfer learning approach, the representations learned through SSL are transferred to target domain and target task, such as object classifi-

cation, detection and semantic segmentation in PASCAL VOC. The patterns learned through any learning algorithm implicitly define a metric on the dataset. Features for similar data samples are similar and dissimilar otherwise. However, the important challenge lies in designing these pre-text tasks such that the model can learn these patterns by itself without involving any human annotations. Some SSL approaches define pre-text tasks through explicit desirable invariances of the metric (5; 6; 7; 3) or such that they implicitly require a good object representation (1; 8; 2).

One of the main challenges this work address is how to compare and understand various pretext tasks. There is only one common evaluation, i.e. transfer features to fine-tune on any supervised object classification, detection or segmentation task, for all the SSL methods. In order to compare different approaches, it is very complicated to use same network architecture to solve both pre-text and target tasks. SSL methods usually use Alexnet (9) as the base architecture. So, one is forced to use the base architecture in the pre-text task which introduces design limitations. For example, pre-text task could exploit any data domains like images, text, sound or video and use different sizes and formats of datasets which may introduce the necessity to use complex architectures.

Recent research has shown better visual representations can be learned through deeper architectures. Complexities of various pretext tasks vary, there is a need to compare difficult and learn-able pretext tasks and arbitrarily deep architectures. To achieve this objective, methods like knowledge distillation (10; 11) can be used to transfer representation from deep complex architecture to a simpler and smaller architecture that could be used to solve target task.



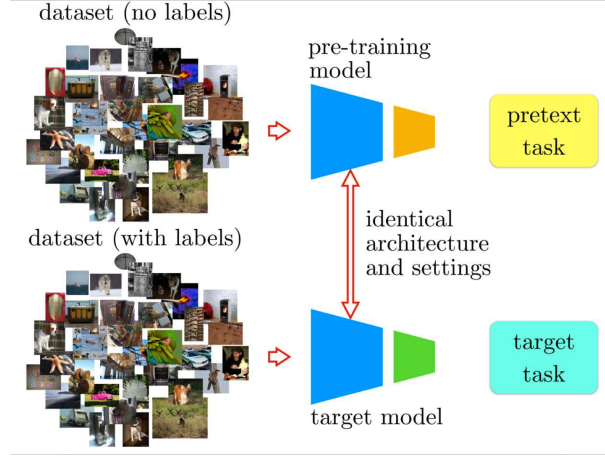


FIG. 1.1. One of the challenges we address in this work is the need to use same architecture for pretraining and fine-tuning. Identical models and settings are used for source task and target task.

In this work, the proposed approach is transfer knowledge by converting learned representation from pretext task into pseudo-labels on an unlabeled dataset. First, the features are learned by solving a pretext task. This SSL training phase could use any complex architecture on any dataset without involving human annotation. Second, we use the learned model to extract features from any dataset of unlabeled images and cluster together (e.g. using K-means) and use cluster centre id as pseudo-labels for unlabeled images. Now, the unlabeled images have labels through this clustering process. This could be seen as transferring knowledge and it is worth to note that this process is completely unsupervised. Third, final representation of a model is learnt by training a simpler/smaller model like (e.g. Alexnet) to classify the images based on the pseudo-labels. Though it seems like a lot of information is being discarded by this knowledge transfer process, if semantically similar images are grouped well together, popular supervised classification algorithms work well performing this task. If the representations are good, features of semantically similar images must be closer

to each other than dissimilar images. That means simple K-means algorithm would group similar images into one cluster and labels obtained are a good estimate of the learned representations.

Pseudo labels can be obtained from any dataset and then the knowledge can be transferred to any model by training to predict on those pseudo labels. Thus our framework offers simplicity and flexibility which enables us to do knowledge transfer from any network to any other network which could be trained on target domain and solve target task. Representations learned from complex architectures and difficult pretext task could be easily transferable through this framework. Another important challenge addressed is comparison and evaluation of various complicated pretext task and complicated architecture (e.g. neural networks, handcrafted features) trained on various datasets under different training settings, on a common platform of a fixed model (Alexnet), data domain (PASCAL VOC) and tasks such as classification and detection. To showcase the effectiveness of the framework, we designed a novel Self-supervised pretext task that is more difficult and uses a deeper model (VGG). This task is an extension of the jigsaw problem (8), where a part of the image is occluded which makes it harder to solve. We call this task as Jigsaw++. This is trained on VGG (12) and then transferred to AlexNet (9) through the framework we proposed. This model achieves state-of-the-art performance on PASCAL VOC classification and detection tasks, ILSVRC'12 classification task. It is worth to note that the model architecture for both the cases is Alexnet.

Thus, to summarize, the following are the main contributions of this thesis:

1. We propose a novel framework to asses incompatible models in SSL context.
2. With extensive analysis, we show deeper neural networks learn better representation on pretext tasks.

3. We transfer knowledge from deeper model to shallower model and boost its performance.
4. We achieve state-of-the-art performance on PASCAL VOC object classification, detection and segmentation tasks.

The remainder of the thesis is organized as follows. Chapter 2, Background and Related Work, discusses self-supervised learning approaches and knowledge distillation. Chapter 3, explains the method of knowledge transfer in detail and describes a more complicated pretext task called Jigsaw++. Chapter 4 describes the datasets used for our experiments. In Chapter 5, 7, reports the numbers for various evaluations on different datasets on tasks such as classification, detection and segmentation. We conclude and discuss future work in Chapter 9, Conclusion.

The research presented in this paper was carried out in collaboration with my research advisor Hamed Pirsiavash and Mehdi Noroozi, Paolo Favaro from University of Bern, Switzerland, and sections of the work have been published as part of the proceedings of the 30<sup>th</sup> International Conference on Computer Vision and Pattern Recognition. We begin the rest of this thesis with a survey of the literature related to the concept of Self-supervised Learning.

## Chapter 2

# BACKGROUND AND RELATED WORK

### 2.1 Convolutional Neural Network

Convolutional Neural Network (CNNs) are deep feed-forward neural networks designed specifically to analyze images. It is a stacked multi-layer perceptron that takes advantage of 2D structure of images. It consists of an input layer, hidden layers and an output layer. Input layer usually takes volume of images, output layer is generally a tensor which could be converted as probability distribution by passing it through appropriate function. Hidden layers consists of various functional layers followed by fully connected layer. It may contain one or more convolutional layers with normalizations, pooling layers and non-linear activation layers. The features obtained by feeding an image through these layers are translation invariant. The main advantage of CNNs is that they are easy to train and their shared weight architecture leads to fewer parameters than fully connected networks with the same number of hidden units. In 2012, (9) used a CNN based model to win ILSVRC 2012 (ImageNet Large-Scale Visual Recognition Challenge). The model became very popular since then due to cheap computing power and less processing time given a large amount of data for training. These models are not only used to solve core computer vision problems like object classification, detection and segmentation, but also used in other

areas like Natural Language Processing (NLP), Speech Processing. Since 2012, people have worked on design of CNN and it is an active area of research. There are various models that have previously achieved high performance on benchmark datasets, such as AlexNet (9), VGG16 (12), InceptionNet, ResNet and more recently DenseNet.

In this report, we have mainly demonstrated the experiments based on two main CNN model architectures, AlexNet and VGG16. The reason to use AlexNet is twofold, (1) SSL pretext tasks generally does not follow straightforward architecture, so the natural choice to use is AlexNet which is easy to train and contains less number of parameters than other deep architectures; (2) All the related research methods employ AlexNet, which forces us to use same model architecture as theirs to do a fair comparison.

### **2.1.1 Alexnet**

AlexNet was designed by Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton in 2012 that competed in 2012 Imagenet challenge and eventually won with a very large margin. It is a simple feed-forward CNN which has 5 convolutional layers which has Local Response Normalization, max-pooling, non-linear activation layers followed by 3 fully-connected layers. It is a shallow neural network that achieved 10.5% top-5 error on ImageNet challenge. It has roughly 60 million parameters where fully connected layers account for 90% of them. We used this model as the main building block in my framework and all the experiments are done using them.

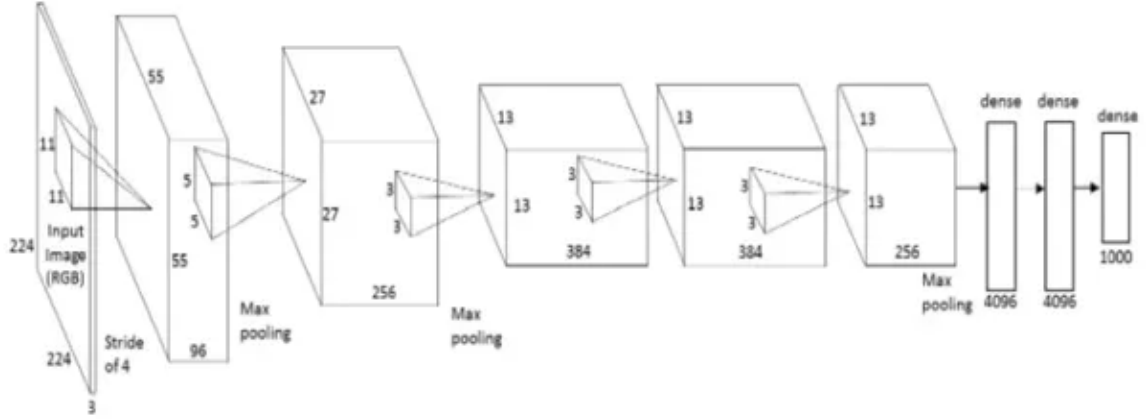


FIG. 2.1. AlexNet architecture

### 2.1.2 VGG-16

VGG was designed by Karen Simonyan and Andrew Zisserman in 2014 that also competed in Imagenet challenge. Like AlexNet, VGG is also a feed-forward CNN but a very deep one. It has 16 layers that consists of convolution, max-pooling and fully connected layers apart from non-linearity layers. It achieved 90% top-5 accuracy on ImageNet. It contains around 140 million parameters. VGG pretrained on ImageNet is generally used as a feature extractor or fine-tuned for other computer vision related tasks. We use this model to solve difficult pretext task and transfer the knowledge from this model to shallow model like AlexNet.

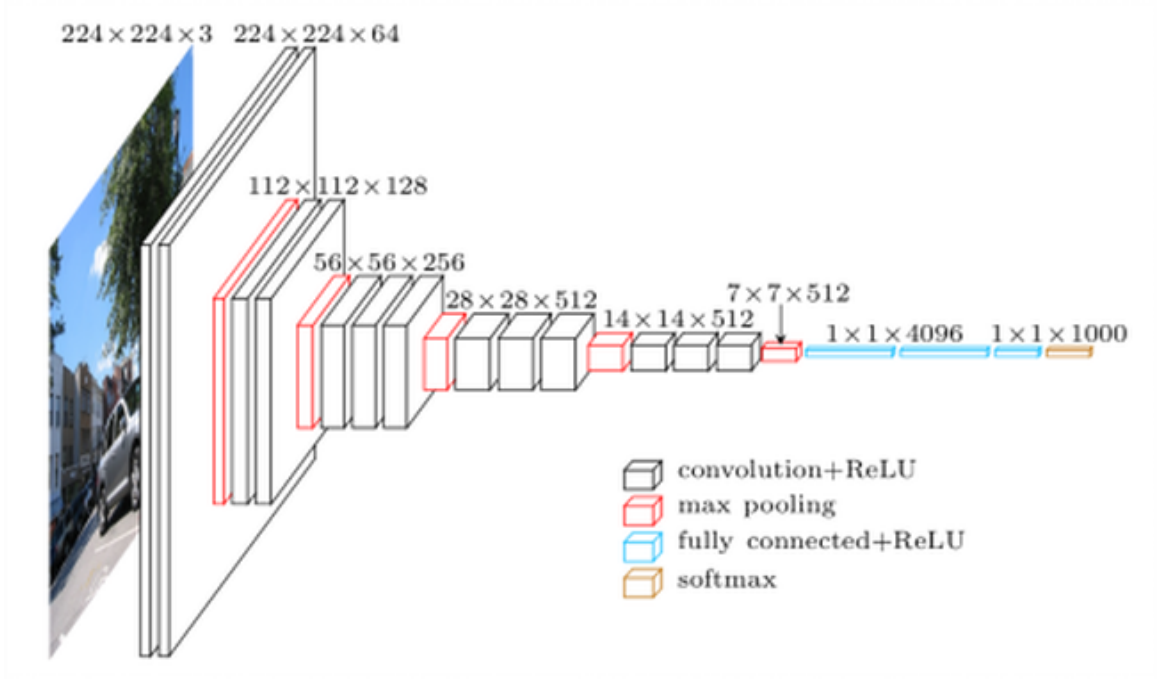


FIG. 2.2. VGG16 architecture

## 2.2 Self-supervised Learning

Self-supervised learning consists of pretext task and some methods design pretext task by reconstructing the data at pixel level from partial observations. This is closely related to denoising autoencoders (13). Colourization problem (2; 14) is an interesting SSL method where given a grayscale, the model is trained to generate a colour image of the same input. A much similar approach was also employed by (15), where they designed a pretext task called as Inpainting. In this method, the model is trained to predict a removed patch on the image. This helps in learning to predict by making use of surrounding information. Wang and Gupta (5) used temporal information in videos to learn representation. Since tracked patches in a video should be semantically similar, they learn a similarity metric to predict similar and dissimilar patches. Signal

reconstruction is also an SSL approach employed by Misra et al. and Brattoli et al. (4; 16) where they train a model to learn to predict the order of video frames. Methods from Doersch et al. (1) uses SSL as a method to predict the spatial relationship between image patches of the image. Noroozi and Favaro (8) design pretext task similar to a jigsaw puzzle solver where the model is trained to predict the permutations of the shuffled patches of the image. By Segmenting an image into foreground and background, Pathak et al. (17) learn signals which also uses the optical flow of adjacent frames from the video. Model is then trained to predict this segmentation. We are interested in making use of information from various data domains like ego-motion (18; 19), or sound (20). This flexibility is not present in SSL paradigm so far.

All the pretext task mentioned in this chapter are assessed through transfer learning and benchmarked on AlexNet architecture and PASCAL VOC datasets. The important question is to how to improve the performance further. Mult-task learning is an option where we could combine more than one SSL pretext task and jointly solve each of the tasks, just like (21; 6), but the computation becomes expensive. The other direction was taken by us is address the need to use the same type of architectures for both pretraining and fine-tuning. Through experimental results, we showcase one could learn rich visual/semantic representations through solving a complex pretext task while using deeper models like VGG16 compared to a shallow model like AlexNet. This framework enables us to use various types of architectures suitable for SSL methods, different data formats and data domains and more complicated pretext tasks. We make the method of Noroozi and Favaro (8) (jigsaw) more challenging by introducing occlusions.



### 2.3 Knowledge Distillation

Knowledge transfer is the crux of our framework where we transfer learned information from one model to another through clustering the features. This method is related to model distillation (10; 22) where they perform distillation by training a target model that learns to predict the probability distribution of a parent or source model. Another way to transfer knowledge is by regressing the neuron activations which is done by (23; 24) which is closely related to our work. Our method is different in a way that where we are trying to preserve the metric of the representation rather than prediction the activations. Dosovitskiy et al. (7) used clustering technique to train & to retrain the same network, however, they do not use clustering for knowledge transfer. Other related work is using clustering to extract labels from unlabeled data for novel object classes (25) or building hash functions (26). None of the works uses methods like ours to transfer knowledge from deeper model to a shallower model. HOG experiment is related to us where they show shallow layers of VGG perform similarly to SIFT features.

## Chapter 3

# METHODS

### 3.1 Knowledge Transfer

The huge advantage and the primary reason SSL methods have become popular is it enables one to use a large amount of data for training that doesn't involve human annotation, which is inexpensive. Basically, in SSL rich visual representations are learned by training a model on pretext task with large amounts of data and then fine-tune it on a supervised target domain and target task for which annotation is already available. This target data would be usually small compared to the pre-training phase. The usual supervised task is object classification and detection (e.g. PASCAL VOC).

The limitation of this process is using the same architecture for pre-training and fine-tuning phase. One cannot use a different architecture for both pre-training and fine-tuning because the weights cannot be transferred between the phases. This becomes an important issue when one uses a more sophisticated and complicated architecture trained on large-scale datasets and then fine-tune it on target task.

The main objective is to not change the architecture on the fine-tuning phase, we need to transfer knowledge between pre-training and fine-tuning phase. This way, we could train pretext task using different architecture compared to the target task,

i.e. fine-tuning phase.

This is compatible with the standard fine-tuning process where we transfer weights up to some convolutional layer and ignoring the final layers. Final layers usually learn task-specific information which could be ignored when weights are used for a different task.

Learning a good representation means features of similar images should be close together than dissimilar images. One way to evaluate this is to do a nearest neighbour search to check if all the images retrieved for a query image is all semantically similar. Nearest neighbours are usually determined using the simple Euclidean distance between the features, which should be good enough to group similar data points. The main objective is to cluster the features learned by the model and get a pseudo label for the images in the dataset. A classifier network is then trained on this dataset with pseudo labels to learn a novel representation.

### **3.1.1 Pre-training**

pre-training is done using Self-supervised learning. Given a pretext task, a particular model architecture and a dataset. The model is trained on the dataset to solve the pretext task as shown in the figure. In this work, we used the convolutional neural network as our architecture. The feature output can be extracted from various level from the model. Refer 3.1 (a) for this step.

### **3.1.2 K-Means**

we use k-means as our clustering algorithm to group the features under specific categories. The distance metric used in this method is Euclidean. Based on empirical results, we fixed the number of cluster centers as 2000. Suppose we extract Imagenet features from this model and perform clustering, the cluster centers should be aligned

with the object categories. Refer 3.1 (b) for this step.

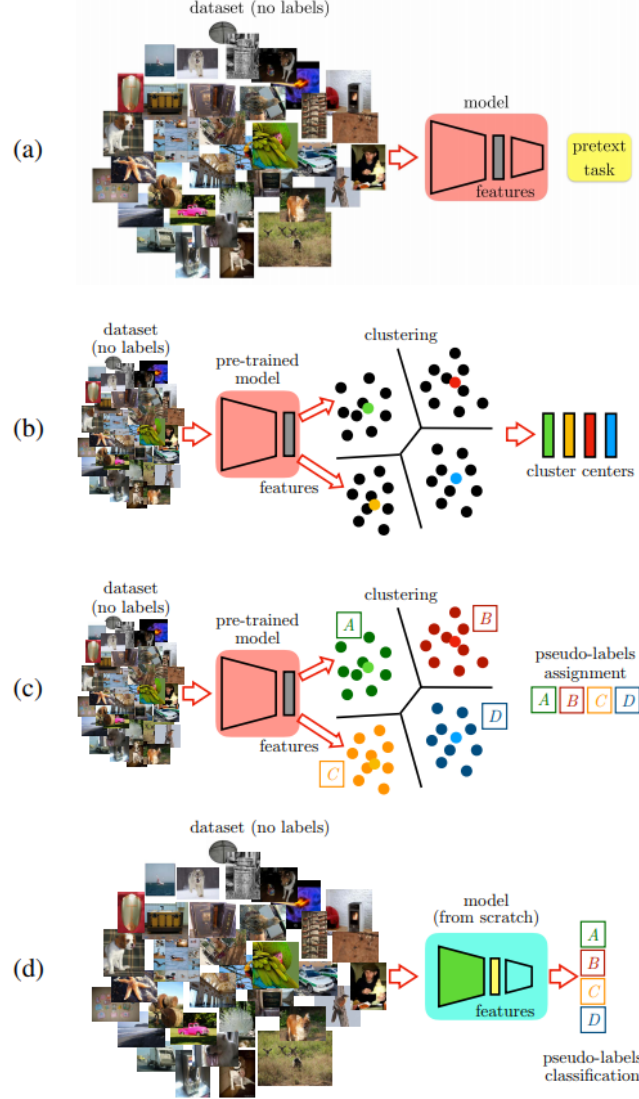


FIG. 3.1. The steps involved in knowledge transfer of our framework. (a) pretraining on pretext task. (b) extract features using the pretrained model. (c) assign pseudo-labels by performing clustering. (d) train a new model from scratch on the dataset to predict the pseudo-labels

### 3.1.3 Pseudo labels

labels for the features can be assigned based on their distance from the cluster centers. Here, we consider cluster centers as pseudo-categories. One flexibility here is, any dataset can be used to obtain labels and it may vary from the dataset used in the clustering step. Refer 3.1 (c) for this step.

### 3.1.4 Cluster Classification

We train a classifier to predict the pseudo-labels obtained from the step above by feeding an image from the dataset used to assign labels. This method learns novel representation in this target model architecture that maps data from pre-trained feature space. Refer 3.1 (d) for this step.

## 3.2 Jigsaw++

Recently, (21; 6) have employed deeper architecture and shown its benefit in SSL methods to learn better representations. But these methods also use the same deep architecture while fine-tuning. This makes it incompatible with previous research methods that use simple models like Alexnet architecture for SSL or fine-tuning. Our objective is to know how can we improve SSL pre-training of Alexnet for PASCAL classification tasks. In our proposed framework, there is no constraint to maintain the same architecture in pre-training and fine-tuning phase, we increased the difficulty of the SSL pretext task and increasing the capacity of the model while still using AlexNet at the fine-tuning phase.

Okanohara et al. (27) built a model to learn text representations by replacing a word randomly and train a model to distinguish it from the original sentence. This approach is used in our jigsaw puzzle pretext task by replacing tiles in the image

puzzle with a random tile from a random image. We term this model as Jigsaw++. The original task is to predict the permutation of tiles from a 3x3 grid of tiles from the same image. In Jigsaw++, we randomly select up to 2 tiles and replace them with the tiles from a random image from the dataset, to make the task more difficult fig 3.2. This could be seen as a form of occlusion and makes the task harder to solve. The model needs to detect the occluding tiles first and then solve the task with the remaining tiles. This is hard because there is a loss of information (the tiles being randomly replaced) from the original image. While we have  $9!$  ways of permuting the tiles, we chose 701 permutations by removing permutations that are closer to each other by calculating Hamming distance between them. We set at least 3 units of Hamming distance.

To prevent the model from learning low-level statistics to detect the random tiles and solve the jigsaw puzzle task, we used gray-scale images 70% of the time in training. In addition to that, we also do mean and std normalization at each image tile. We use Alexnet and VGG16 net model architectures for Jigsaw++ task. The reason behind using a deeper architecture is to equip the model with the larger capacity to handle the complexity of the Jigsaw++ task and to learn better visual representations from the dataset. The pipeline is shown in fig 3.2. We train the model with new Jigsaw++ pretext task, then extract features and perform clustering, assign pseudo-labels and train Alexnet to classify the pseudo-labels. This way, we show knowledge of transfer is done. We obtain significant performance gain by using VGG16 with the Jigsaw++ task in fine-tuning on target dataset (PASCAL). This clearly shows that deeper architecture leads to better representations.

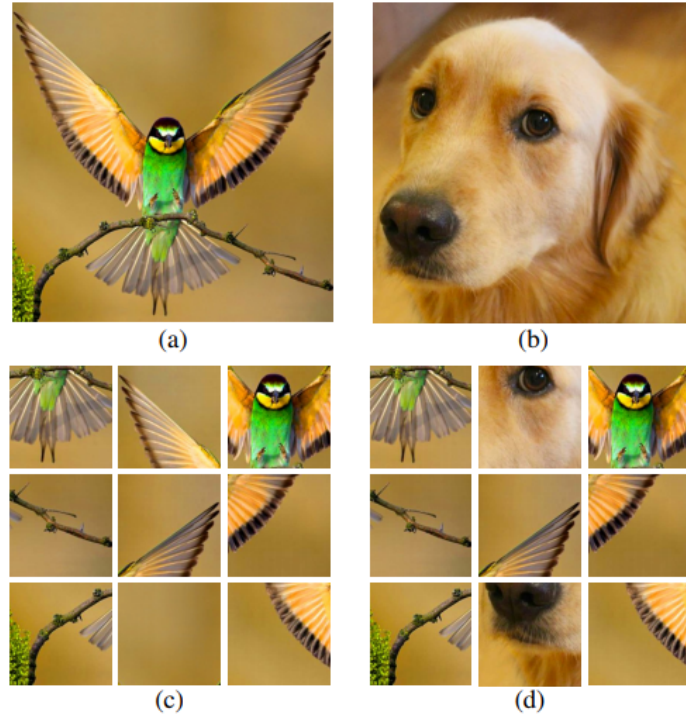


FIG. 3.2. A dataset example for Jigsaw++ pretext task where we divide an image into 9 tiles and randomly replace up to 2 tiles from a random image.

## Chapter 4

# DATASET

This chapter describes the dataset we have used in this project. For classification, we use the Imagenet ILSVRC dataset, Places, and PASCAL VOC dataset. For object classification and detection, we use the Pascal VOC (28) dataset to fine-tune the pretrained model.

### 4.1 ImageNet

The Imagenet is huge collection of dataset that has over 1.3M images gathered from multiple sources such as Google, Yahoo, Flickr etc. This has been an important dataset that benchmarked almost all the models. These images have been annotated by humans using Amazon's crowd-sourcing platform for human intelligence tasks: Mechanical Turk. Imagenet has 1000 categories. Imagenet is used every year for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) that evaluates different object detection and image classification algorithms. We used the train split for training and val split for testing. We used Imagenet for pretraining the model, feature extraction for clustering, cluster classification, evaluation of Linear and non-Linear classifiers.



## 4.2 PASCAL VOC

The Pascal Visual Object Challenge (VOC) is another dataset for object recognition and segmentation task. It was started in the year 2005. The object classes of Pascal dataset have been organized into the following categories.

- **Person:** person
- **Animal:** bird, cat, cow, dog, horse, sheep
- **Vehicle:** airplane, bicycle, boat, bus, car, motorbike, train
- **Indoor:** bottle, chair, dining table, potted plant, sofa, tv/monitor

Pascal VOC dataset has been annotated for object detection, specifying localization information for each object within the image. It provides the coordinates of the center of the image, height and width of objects which can be used for training object detection systems.

The distributions of images and objects by class are approximately equal across the training/validation and test sets. In total there are 9,963 images, containing 24,640 annotated objects. In table 4.1 given below, we enlist the training and testing images we have used according to each individual class.

task	aero	bicycle	bird	boat	bottle	bus	car	cat	chair	cow
training	908	795	1095	689	950	607	1874	1417	1564	444
testing	204	239	282	172	212	174	721	322	417	127
task	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
training	738	1707	769	771	6095	772	421	736	805	831
testing	190	418	274	222	2007	227	97	223	259	229

Table 4.1. Images by category in Pascal VOC Dataset

### 4.3 PLACES

Places is a scene recognition dataset. Scene recognition is an important task in computer vision, allowing defining a context for object recognition. Places dataset has 205 scene categories and 2.5 millions of images with a category label. Here, we used a random sampling of 1 million images which is constant across all our experiments. We use this dataset for linear classification of pretrained models, feature extraction from pretrained model for clustering, and cluster classification. We used this dataset because it does not have biases like ImageNet where classes have equal number of images.

## Chapter 5

# EXPERIMENTS

In Chapter 3, we described the purpose of transfer knowledge and the methodology of it being used in this framework. We also described about the new pretext task that is more complex than original jigsaw puzzle pretext task. In this Chapter, we describe the experiments conducted for knowledge transfer, pretraining using pretext task on various models and fine-tuning on benchmark datasets.

We conducted extensive evaluations on knowledge transfer methods and the Jigsaw++ task. We used various datasets to show the performance of the frameworks. Several transfer learning benchmarks like PASCAL VOC fine-tuning, Non-Linear classification on Imagenet, Linear classification on Places dataset and Imagenet dataset.

In addition to these experiments, we also conducted ablation studies to the effect of the number of clusters in cluster classification, different datasets used for clustering and generation of pseudo-labels between one dataset to another to transfer the knowledge. Performing pre-training on Jigsaw++ pretext task yields features that achieve the best performance on benchmark datasets by fine-tuning and better than the current self-supervised learning methods. It is worth to note that all the convolution layer weights from AlexNet model from pretext task is transferred to target AlexNet and dense layers (fully-connected layers) are initialized randomly. The knowledge

transfer is experimented with various models and self-supervised learning methods but the cluster classification model is fixed as AlexNet. This is applicable for all the experiments we conducted in this work.

### 5.1 Jigsaw++ trained with VGG16

We used a deeper architecture like VGG16 on pretext task. This method obtains state-of-the-art performance on all the benchmark datasets. We also show that deeper architecture trained on pretext task helps in learning richer representations.

### 5.2 Jigsaw++ trained with AlexNet

We experimented the difficult Jigsaw++ task with AlexNet architecture and evaluated on all benchmark dataset. The performance did not drop while using a smaller model like AlexNet. This is evident that a pretext task could be reduced to a classification task i.e, through our framework where we transfer knowledge as pseudo-labels which in-turn is a classification task. But only if the model has learnt a proper similarity metric during pretraining.

### 5.3 Context Prediction

We trained the method of Doersch et al. (1) on AlexNet model to show how other self-supervised learning methods can be plugged into our framework. The experimental results show this method achieved better results than theirs. Originally, the context prediction method uses batch normalization (29) during pretraining. We found that using batch normalization during pretraining affects the learning during fine-tuning since the settings during fine-tuning is different. Krähenbühl et al. (30) rescaled the weights and showed it boosted performance during fine-tuning. Our

method achieves better performance than (30).

#### 5.4 Histogram of Oriented Gradients

We used HOG features which are hand-crafted, to extract image features instead of using any self-supervised learning methods. To much of our surprise, this method achieved high performance in fine-tuning on classification in PASCAL VOC, outperforming all other self-supervised learning methods till date.

#### 5.5 Random Pseudo-labels

We conduct an experiment using AlexNet model randomly initialized and extract features to do clustering and obtain pseudo-labels. Zhang et al. (31) showed one could train a model to predict random labels. During, cluster classification, the network is able to achieve good performance on training set but badly on the validation set. The accuracy on the validation set is close to random, which is expected. Then we fine-tuned this model on PASCAL VOC classification and found that the performance is same as training a model from scratch.

#### 5.6 Knowledge Distillation

The method used by (10) cannot be applied directly to our framework, so we decided to train a student network to regress conv4-5 layers of the teacher network. In this case, we use AlexNet for both teacher and student network. In this method, the teacher network archives 69.8% while the student network obtains 58.5% on PASCAL VOC Classification task. Then, we use Jigsaw++ trained on VGG16 as the teacher and use AlexNet as the student. This method did not provide a significant performance boost and the results can be seen in Table 3.

## 5.7 Implementation Details

The features we extracted for all the experiments in this framework are from the conv4 layer and max-pool them to 5x5x384 for AlexNet and 4x4x512 for VGG16 network architecture. We implemented standard K-means algorithm with the number of clusters  $k=2000$ . It takes up to 4 hours to cluster 1.3M images and run on single Titan X GPU. The hyper-parameters used in the experiments are identical to standard AlexNet and ImageNet classification settings to do cluster classification.

## Chapter 6

# ABLATION STUDIES

In the previous chapter, we described all the experiments being carried out in this framework. We showed various model, different datasets and different self-supervised learning methods that can be plugged into this framework that achieved best performance on benchmark datasets. In this chapter, we do various ablation studies to completely understand and evaluate what the model has learnt and study the impact of the number of clusters, different data domains.

In all the evaluations mentioned in this chapter, we used AlexNet architecture pretrained on Jigsaw++ pretext task with ImageNet as the dataset. We used the train split of the dataset without labels for pretraining. The pseudo-labels obtained after k-means clustering is also from ImageNet dataset unless otherwise stated. The final stage where the knowledge is transferred by classification of pseudo-labels are done using AlexNet architecture. For fine-tuning, we use PASCAL VOC object classification dataset.

### 6.1 Impact of Number of Clusters

Usually, user provides the number of clusters in k-means algorithm and this is determined by running k-means with various number of number of clusters and

choose the number which obtains optimal performance. Too few clusters may be not be very discriminative because lot of dissimilar datapoints could be grouped together. Increasing the number of clusters to a high value may not generalize well as some of the cluster centers may not contain any datapoints. Since we convert the cluster to a pseudo-label, it becomes the number of classes or objects the final model is going to be trained to discriminate. Hence, we can say that the network trained on very few pseudo-labels may be worse than a network trained on large number of pseudo-labels for that particular object category. We show this as an analogy to the experiment done on ImageNet labels (32). On Table 6.1, we could see the correlation between the performance of the network with the number of clusters. As the number of clusters increases, number of datapoints in each clusters decreases, which causes the network to overfit and that leads to decrease in its performance.

Table 6.1. **Impact of the number of clusters.**

<b>number of clusters</b>	<b>500</b>	<b>1000</b>	<b>2000</b>	<b>5000</b>	<b>10000</b>
<b>mAP on voc-classification</b>	69.1	69.5	69.9	69.9	70.0

## 6.2 Impact of Data Domain

This is one of the studies that showcases our framework’s flexibility in handling different data domains. Our framework enables us to use different data domains during pre-training phase, perform cluster on another data domain and extract features on a third data domain. The result of the study of some of the data domains to showcase the biases between them are shown in Table 6.2. In one of the experiments shown in the table, we learn cluster centers on conv4 features of Jigsaw++ extracted on Places dataset, obtain pseudo-labels by running clustering on ImageNet and then



train it on AlexNet for cluster classification. We see a small reduction in performance which shows that our clustering methodology does not depend on any biases inherent in the ImageNet dataset.

Table 6.2. **Impact of the cluster and pseudo-labels data domains.**

<b>clustering on pseudo-labels on</b>	ImageNet	ImageNet	Places
	ImageNet	Places	ImageNet
<b>mAP on voc-classification</b>	69.9	68.4	68.3

## Chapter 7

# TRANSFER LEARNING EVALUATION

Transfer learning is the standard evaluation method for Self-supervised learning algorithms. These are done to evaluate how good the representations are and whether or not it is transferable to another task. We conduct the evaluation experiments on PASCAL VOC for object classification, detection and segmentation. We also apply our knowledge transfer method to some of the SSL methods under relevant settings. We performed evaluations for knowledge transfer for Context (1), Context-ColorDropping (1), Context-ColorProjection (1), Jigsaw++, and HOG (33).

### 7.1 PASCAL VOC Fine-tuning

PASCAL VOC is a common benchmark for all SSL methods and our evaluations are done by fine-tuning on this dataset. We use the framework of (30) and Fast-RCNN (34) to conduct all the experiments. Semantic Segmentation task is also reported on PASCAL VOC 2012 dataset using the framework of (35). In most recent SSL paper, the detection task settings are not the same ones used for SLL methods. SLL methods use multi-scale fine-tuning for 150K iterations with the learning rate of 0.001 and it is reduced by 10 every 50K iterations. We fine-tuned the supervised learning weights with these settings and achieved 59.1% and 59.9% mAP with multi-

scale and single scale test. We use the same setting for all the experiments and specifically use the mentioned accuracy as baselines. We freeze the first convolutional layer in all experiments as it was the case in Fast-RCNN. Table 7.1, we use "CC+" when the knowledge-transfer method is used and "vgg-" when VGG16 is used in pre-training. Methods in Table 7.1 use AlexNet for cluster classification and fine-tuning. We apply HOG features on ImageNet and observe a very high performance in all three tasks. We did not see significant improvement when we used same architecture for the pretext and fine-tuning tasks. When pretext task is training using VGG16, there was a significant improvement up to 2.6% in classification, 1.6% in detection and 2.6% in semantic segmentation tasks. This is the state-of-the-art performance on all the tasks.

Table 7.1 shows evaluation of SSL methods on classification, detection and semantic segmentation. All rows use AlexNet architecture at finetuning. "vgg-" means VGG16 architecture is used before knowledge transfer. For "CC+vgg-Jigsaw++", we train Jigsaw++ using VGG16, cluster, and then train AlexNet to predict clusters. We also transfer using (24) in "(24)+vgg-Jigsaw++". For "CC+HOG", we cluster bag of words of HOG and predict it with AlexNet, so it is not completely unsupervised since HOG is hand-crafted. Surprisingly, this outperforms most SSL algorithms.

Method	Ref	Class.	Det.		Segm.
			SS	MS	
Supervised (9)		79.9	59.1	59.9	48.0
CC+HOG (33)		70.2	53.2	53.5	39.2
Random	(15)	53.3	43.4	-	19.8
ego-motion (19)	(19)	54.2	43.9	-	-
BiGAN (36)	(36)	58.6	46.2	-	34.9
ContextEncoder (15)	(15)	56.5	44.5	-	29.7
Video (5)	(30)	63.1	47.2	-	-
Colorization (2)	(2)	65.9	46.9	-	35.6
Split-Brain (37)	(37)	67.1	46.7	-	36.0
Context (1)	(30)	55.3	46.6	-	-
Context (1)*	(30)	65.3	51.1	-	-
Counting (3)	(3)	67.7	51.4	-	36.6
WatchingObjectsMove (17)		61.0	-	52.2	-
Jigsaw (8)	(8)	67.7	53.2	-	-
Jigsaw++		69.8	55.5	55.7	38.1
CC+Context-ColorDrop (1)		67.9	52.8	53.4	-
CC+Context-ColorProjection (1)		66.7	51.5	51.8	-
CC+Jigsaw++		69.9	55.0	55.8	40.0
(24)+vgg-Jigsaw++		70.6	-	-	38.0
CC+vgg-Context (1)		68.0	53.0	53.5	-
CC+vgg-Jigsaw++		<b>72.5</b>	<b>56.5</b>	<b>57.4</b>	<b>42.6</b>

Table 7.1. PASCAL VOC Classification.

## 7.2 Linear Classification

We conduct experiments to evaluate SSL methods for Linear classification on the features extracted from AlexNet at different layers (37). We used both ImageNet (38) and Places (39) dataset in this experiment. Table 7.2 reports the classification performance on ImageNet dataset and Table 7.2 reports the classification performance on Places dataset. Jigsaw++ model achieves comparable performance with other state-of-the-art SSL methods. Applying knowledge transfer is seen helpful in the transferred model CC+Jigsaw++. Pretraining on deeper architecture like VGG16 boosts the performance. HOG features did not perform well compared to its performance on PASCAL VOC. We observe the performance of pretraining with CC+vgg-Jigsaw++ is close to supervised pretraining with ImageNet labels.

Method	conv1	conv2	conv3	conv4	conv5
Places labels (39)	22.1	35.1	40.2	43.3	44.6
ImageNet labels (9)	22.7	34.8	38.4	39.4	38.7
CC+HOG (33)	20.3	30.0	31.8	32.5	29.8
Random	15.7	20.3	19.8	19.1	17.5
Context (1)	19.7	26.7	31.9	32.7	30.9
Jigsaw (40)	23.0	31.9	35.0	34.2	29.3
Context encoder (15)	18.2	23.2	23.4	21.9	18.4
Sound (20)	19.9	29.3	32.1	28.8	29.8
BiGAN (36)	22.0	28.7	31.8	31.3	29.7
Colorization (2)	16.0	25.7	29.6	30.3	29.7
Split-Brain (37)	21.3	30.7	34.0	34.1	32.5
Counting (3)	<b>23.3</b>	33.9	36.3	34.7	29.6
Jigsaw++	22.0	31.2	34.3	33.9	22.9
CC+Jigsaw++	22.5	33.0	36.2	36.1	34.0
CC+vgg-Jigsaw++	22.9	<b>34.2</b>	<b>37.5</b>	<b>37.1</b>	<b>34.4</b>

Table 7.2. **Places classification with a linear classifier.** We use the same setting as in Table 7.3 except that to evaluate generalization across datasets, the model is pre-trained on ImageNet (with no labels) and then tested with frozen layers on Places (with labels).

Method	Ref	conv1	conv2	conv3	conv4	conv5
Supervised (9)	(37)	19.3	36.3	44.2	48.3	50.5
CC+HOG (33)		16.8	27.4	20.7	32.0	29.1
Random	(37)	11.6	17.1	16.9	16.3	14.1
Context (1)	(37)	16.2	23.3	30.2	31.7	29.6
ContextEncoder (15)	(37)	14.1	20.7	21.0	19.8	15.5
BiGAN (36)	(37)	17.7	24.5	31.0	29.9	28.0
Colorization (2)	(37)	12.5	24.5	30.4	31.5	30.3
Split-Brain (37)	(37)	17.7	29.3	35.4	35.2	32.8
Counting (3)	(3)	18.0	30.6	34.3	32.5	25.7
Jigsaw++		18.2	28.7	34.1	33.2	28.0
CC+Jigsaw++		18.9	30.5	35.7	35.4	32.2
CC+vgg-Jigsaw++		<b>19.2</b>	<b>32.0</b>	<b>37.3</b>	<b>37.1</b>	<b>34.6</b>

Table 7.3. **ImageNet classification with a linear classifier.** Every column shows the top-1 accuracy of AlexNet on the classification task. The learned weights from conv1 up to the displayed layer are frozen. The features of each layer are spatially resized until there are fewer than 9K dimensions left. A fully connected layer followed by softmax is trained on a 1000-way object classification task.

Method	Ref	conv1	conv2	conv3	conv4	conv5	fc6	fc7
Supervised (9)	(9)	57.3	57.3	57.3	57.3	57.3		
Random	(8)	48.5	41.0	34.8	27.1	12.0	-	-
Video (5)	(8)	51.8	46.9	42.8	38.8	29.8		
BiGAN (36)	(36)	55.3	53.2	49.3	44.4	34.9	-	-
Counting (3)		54.7	52.7	48.2	43.3	32.9	-	-
Context (1)	(8)	53.1	47.6	48.7	45.6	30.4	-	-
Jigsaw (8)	(8)	54.7	52.8	49.7	45.3	34.6	-	-
Jigsaw++		54.7	52.9	50.3	46.1	35.4	-	-
CC+-vgg-Context		55.0	52.0	48.2	44.3	37.9	29.1	20.3
CC+Jigsaw++		55.3	51.9	51.4	47.6	41.1	33.9	25.9
CC+vgg-Jigsaw++		<b>55.9</b>	<b>55.1</b>	<b>52.4</b>	<b>49.5</b>	<b>43.9</b>	<b>37.3</b>	<b>27.9</b>

Table 7.4. **ImageNet classification with a non-linear classifier.** Every column shows top-1 accuracy of AlexNet. The learned weights from conv1 up to the displayed layer are frozen. The rest of the network is randomly initialized and retrained.

### 7.3 Nonlinear Classification

In this section, we describe the experiments done on SSL methods for Non-linear classification by freezing various layers initialized with weights transferred from the pretrained model and training the rest of the network from scratch. The difference between these experiments to the previous experiments is that here we use nonlinear classifier of the final layers of AlexNet. The purpose of this experiment is to evaluate the alignment between the pseudo-labels obtained from cluster classification and the ground truth labels. Table 7.2 reports the results. The performance of CC+vgg-Jigsaw++ is significant compared to other SSL methods.



## Chapter 8

# VISUALIZATIONS

We illustrate the filters of the cluster classifier network in Figure 8.1. This figure clearly shows the impact of pretrained network on the cluster classifier. We could see our method is consistent with "Color Dropping" method of (1) after knowledge transfer since the model did not see any color images in the pre-training phase. Figure 8.2 shows some of the images sampled from the clusters used in the training of CC+vgg-Jigsaw++. Each row corresponds to images closest to the center of a single cluster. Ideally, for high-quality representations, we expect images from the same category on each row.

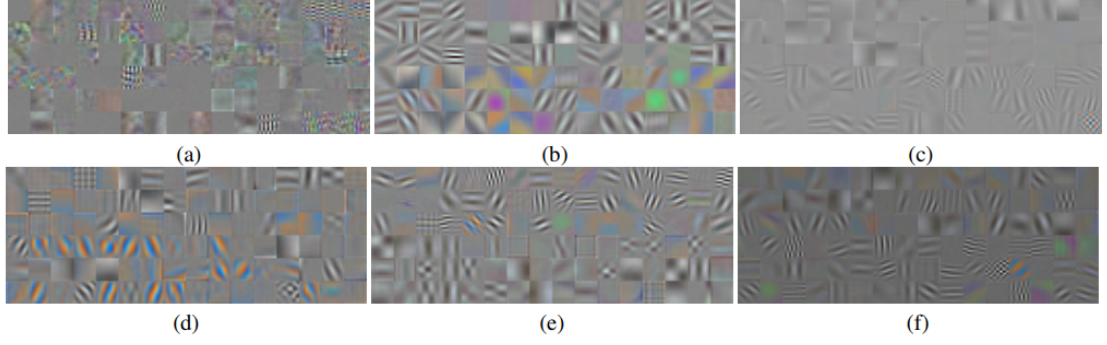


FIG. 8.1. Visualizations of conv1 weights, [a] randomly initialized AlexNet network, [b] HOG features, [c] Doersch et al. trained with color dropping, [d] Doersch et al. trained with color projection, [e] the task jigsaw++ trained on AlexNet, [f] the task jigsaw++ trained on VGG16.

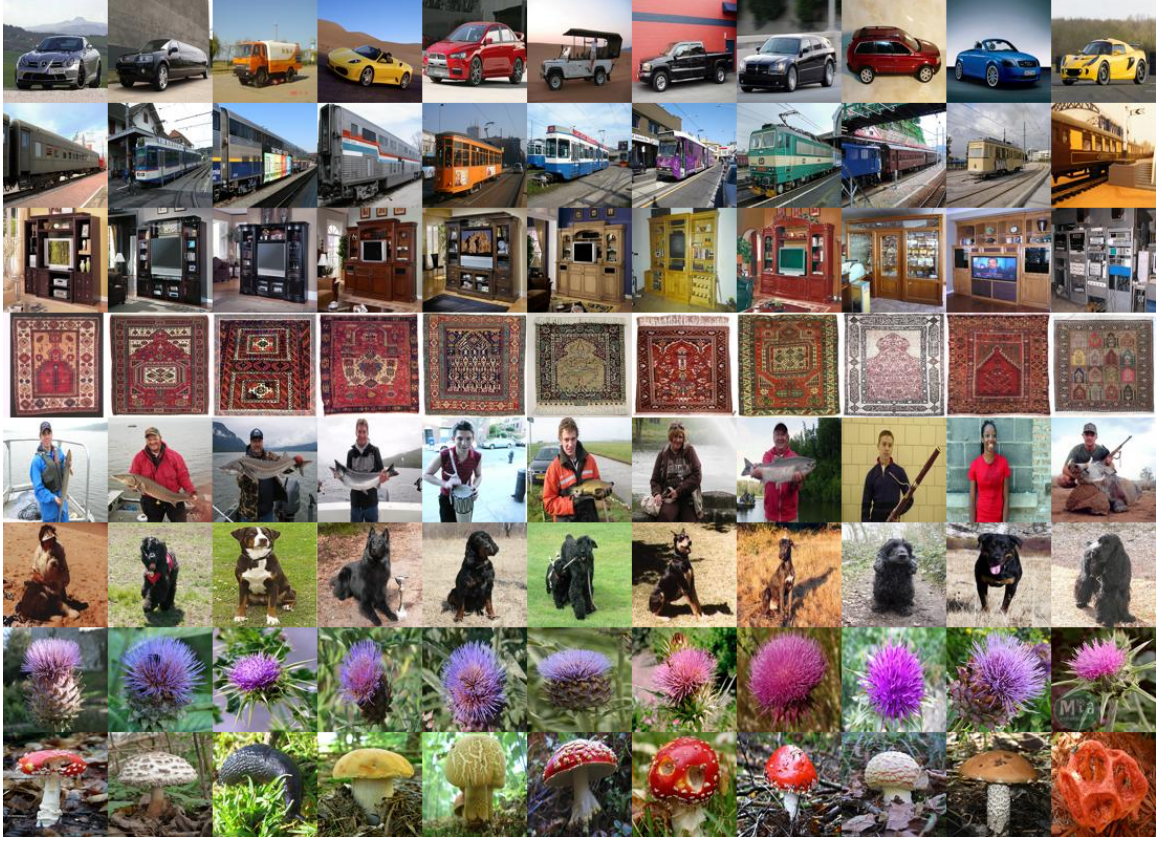


FIG. 8.2. Some cluster samples used to train  $\text{jigsaw}++^{vcc}$ . Each row shows the 11 closest images to their corresponding cluster center.

## Chapter 9

# CONCLUSION

Self-supervised learning is becoming very popular in computer vision research field because of the fact that it uses unlabeled data which is abundant and it is very inexpensive to collect. It can be easily scaled and trained on a very large corpus of data, unlike Supervised learning methods where annotated data is less and very expensive to collect. The main constraint in SSL methods is to use the same architecture in both pretraining stage and fine-tuning stage. Naturally, there is a limitation to use a large amount of training data because of the model capacity of the final task. In this work, we proposed a novel framework that separates the pretraining from fine-tuning part of transfer learning process. This gives the flexibility to plug various models, use different data domains and boosts the final performance through our knowledge transfer method based on clustering the features using a simple k-means algorithm. We also showed that we could learn rich representations using deeper networks in pretraining stage and thus successfully transferred the knowledge to a shallow model which achieved state-of-the-art performance on common benchmark datasets like ImageNet, PASCAL VOC object classification and detection tasks.

## REFERENCES

- [1] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.
- [2] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *ECCV*, 2016.
- [3] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *ICCV*, 2017.
- [4] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, 2016.
- [5] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015.
- [6] Xiaolong Wang, Kaiming He, and Abhinav Gupta. Transitive invariance for self-supervised visual representation learning. In *ICCV*, 2017.
- [7] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *PAMI*, 2014.
- [8] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.
- [9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012.

- [10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS*, 2014.
- [11] Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *KDD*, 2006.
- [12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.
- [13] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2006.
- [14] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *ECCV*, 2016.
- [15] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.
- [16] B. Brattoli, U. Büchler, A. S. Wahl, M. E. Schwab, and Björn Ommer. Lstm self-supervision for detailed behavior analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [17] Deepak Pathak, Ross Girshick, Piotr Dollr, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. *arXiv preprint arXiv:1612.06370*, 2016.
- [18] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *ICCV*, 2015.
- [19] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *ICCV*, 2015.

- [20] Andrew Owens, Jiajun Wu, Josh H. McDermott and William T. Freeman, and Antonio Torralba. Ambient sound provides supervision for visual learning. In *ECCV*, 2016.
- [21] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. *ICCV*, 2017.
- [22] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NIPS*, 2014.
- [23] Yuxiong Wang and Martial Hebert. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*, 2016.
- [24] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017.
- [25] Yu-Xiong Wang and Martial Hebert. Learning from small sample sets by combining unsupervised meta-training with cnns. In *NIPS*, 2016.
- [26] Yu-Xiong Wang, Liangke Gui, and Martial Hebert. Few-shot hash learning for image retrieval. In *ICCVW*, 2017.
- [27] Daisuke Okanohara and Junichi Tsuji. In *ACL*, 2007.
- [28] Mark Everingham, S M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. 2014.
- [29] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

- [30] Philipp Krähenbühl, Carl Doersch, Jeff Donahue, and Trevor Darrell. Data-dependent initializations of convolutional neural networks. In *ICLR*, 2016.
- [31] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.
- [32] Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros. What makes imagenet good for transfer learning? In *NIPS LSCVS Workshop*, 2016.
- [33] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.
- [34] Ross Girshick. Fast r-cnn. In *ICCV*, 2015.
- [35] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.
- [36] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017.
- [37] Richard Zhang, Phillip Isola, and Alexei A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, 2016.
- [38] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015.
- [39] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.



- [40] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *arXiv preprint arXiv:1603.09246*, 2016.

