University of California

Los Angeles

Visual Concept for Foreground and Background Objects Using Deep Lab

A thesis submitted in partial satisfaction of the requirements for the degree Master of Science in Statistics

by

Seyed Ali Mousavi

2018

ABSTRACT OF THE THESIS

Visual Concept for Foreground and Background Objects Using Deep Lab

by

Seyed Ali Mousavi

Master of Science in Statistics

University of California, Los Angeles, 2018

Professor Ying Nian Wu, Chair

We are interested in studying the application of computer vision and visual clusters on creating both foreground and background objects. This is motivated by the observation that deep neural networks learn context as part of the objects. Therefore, by having a dictionary of parts for both foreground and background objects we can capture objects in occluded settings and separate the foreground from the background. Our work makes further contribution by having one network for both foreground and background objects rather than having multiple networks for multiple set of objects. We have demonstrated that a combination of the right network architecture and clustering algorithm can create visual concepts that can be used for both foreground and background objects.

The thesis of Seyed Ali Mousavi is approved.

Alan Loddon Yuille

Hongjing Lu

Yingnian Wu, Committee Chair

UNIVERSITY OF CALIFORNIA, Los Angeles

2018

Dedication

I dedicate this to my family: Mom & Dad, Mohammad, Zeinab and Zahra. To my little nephew and nieces: Fatemeh, Mohammad Ali, Mohammad Hossien, Mohammad Bagher, and Noora. Also to my friends who always supported me, especially Jennifer, Lisa and Renee. Last but not least Allah Almighty.

Table of Content

1 **Introduction** 1

2 **Related Work** 4

3 **Deep Network** 6

3.1 Backpropagation 6

4 **Training Network.** 9

4.1 Data Set and Preparation 9

4.2 Network Architecture 11

4.3 Deep-Lab Architecture 11

4.4 Deep-Lab vs VGG 12

5 **Pascal Context Dataset Results**. 17

5.1 Training and Results 17

6 **Clustering** 20

6.1 Image Selection 20

6.2 Layer Feed Network 22

6.3 Clustering 23

6.4 K-Mean 23

6.5 K-Median. 24

6.6 Advantage of K-median Over K-means 24

6.7 Number of Clusters 24

7 **Evaluation** 26

7.1 Prediction Step. 26

7.2 Splitting the Data 27

7.3 The Challenge with IOU 27

7.4 Our Metric 28

7.5 Results and Discussion 30

8 Conclusion and Future Work 33

List of Figures

1 Figure 1 10

2 Figure 2 13

3 Figure 3 15

4 Figure 4 21

5 Figure 5 22

6 Figure 6 27

7 Figure 7 29

List of Tables

1 **Table 1** 18-19

2 **Table 2** 32

List of Equations

1 **Equation 1** 7

2 **Equation 2** 7

3 **Equation 3** 7

4 **Equation 4** 7

5 **Equation 5** 14

6 **Equation 6** 22

Acknowledgments

Foremost, I want to thank Professor Yuille for accepting me as his student. Throughout the years he was always helpful, caring, passionate, patient, supportive and guided me in my research. I learned more under you than I ever envisioned. Thank you for not doubting me. THANK YOU.

Secondly, I want to thank Professor Wu for accepting me and guiding me once professor Yuille left. If it were not for him, I would have `never had finished this thesis.

I also want to thank my labmates for their support: Vittal Premachandran, Roozbeh Mottaghi, John Flynn, George Papandreou, Boyan Bonev, Jianyu Wang, Danniil Pakhomov, Liang-Chieh (Jay) Chen, Ehsan Jahangiri, Xiaochen Lian, and Xienjie Chen.

Adam Tableman for teaching me basically everything I know: math, programming, research, behaving as a graduate student, life, politics and career advice.

Zeinab and Zahra Mousavi for always being there for me, giving advice, double checking my approach and yelling at me.

Last but not least Glenda Jones, who was like a second mom. Always had my back. Quick for giving me lectures and hugging me right after. Thank you for keeping it real and showing love.

Thank You everyone.

Chapter 1

Introduction

The landscape of computer vision and image classification has evolved over the past few years, mainly because of the contribution of the Deep Neural Networks(DNN) and Graphical Programming Units(GPU)[5]. In addition, many large datasets has been released over the past few years including PASCAL, MS-COCO, and ImageNet. The combination of large dataset and heavy computations power have gain improvements from the traditional approaches such as HOG and Bag of words.

One of the benefits of DNNs is its significant increase in accuracy, precision and recall. The biggest downfall of this approach is our lack of understanding of what the algorithms learn and what each neuron or sets of neurons learn and get activated by. As a result, most researches how been done on macro scale, changing the architecture of DNNs, and many trial and errors approach. Moreover, most of the research on the field is focus on objects of interest and foreground regions. On the other hand, useful information can be contained by learning parts of objects and the background objects in images. This becomes particular an issue in terms of handling occlusion and understanding contexts. [9] demonstrated that background images and visual context can help with image recognition.

Some approaches have been done by training deep neural networks either purely on parts or background images. Although the results are promising, this methods requires separate network training, and have not been combined with foreground object trained networks in any useful matters.

Moreover, understanding and having a dictionary of parts could be useful for many applications. A combination of series of parts can be use to train the objects for context free detection. Dictionary of parts can further enable us to improve detection in highly occluded images.

We have approached these problems by training one network for both foreground and background objects. Then analyzing the activation of different layers of DNNs that have been trained and clustering them into similar patterns, then creating a new metric to score and use as feedback. By doing so we have achieved the highest accuracy in the Pascal Context Dataset and able to have a deeper understanding of what each neuron studies.

First we will train network on set of foreground and background objects. Then we will take the pre-train network and feed it labeled parts of objects. At each layer, particularly at layer 4, we will save the sets of neurons, and their corresponding weights, that have been activated by the image. Thirdly we cluster the neurons, fourthly we combine these clusters into a dictionary that can be chosen to predict an image, lastly we developed a metric to effectively measure the accuracy of our prediction.

Our approach has few advantages, firstly it does not require us to train a new set of DNN for each set of objects. Moreover, this algorithm can accurately predict the object,

even when part, rather than the whole object is feeded to it. Furthermore, by adding more trained objects and context, we improved the detection ability of the network.
The remainder of this paper is organized as follows. Chapter 2 briefly introduces related work. Chapter 3 gives a brief introduction to Deep Neural Network, Chapter 4 and Chapter 5 discusses our training and results. Chapter 6 discusses clustering, Chapter 7 discusses the different dictionary of objects and chapter 8 summarize this paper and propose future work in Chapter 5.

Chapter 2

Related Work

There has been some studies on Mid-level patches and visual concept. [7] used CNN features to discover mid-level patterns that were used for object classification. [12] and [19] used mid-level neuron activation for object part detection, however their focus was on a single neuron. Activations from intermediate layers of the CNN to find object parts, which they then used for fine-grained object classification. But they assumed that a single filter from an intermediate layer can detect parts. [1] Tried doing object classification using multiple filters for object classification, however their work used SVM training, which differs than out unsupervised method.

The closest work was done by Wang[14][15][16][17][18]. Wang proposed a new approach to finding partial object representations while dealing with DNN internal states. These new representations, dubbed as “visual concepts”, are extracted through using clustering technique. The DNN was specifically trained on 6 sets of Pascal objects. For each object the authors used K++, k-mean algorithm to cluster the neurons. This process provided the authors with a dictionary containing all detected visual concepts per each layer of DNN.

An important finding by the authors, was their ability to prove the connection between the visual concepts and the characteristically separate parts of the objects (represented by the patches of the original image), with a high enough degree of correspondence between the two.

Wang showed that visualization and clustering of similar items are effective. However their study was limited to rigid objects of PASCAL 3D+ objects, and used part-label for classification. Our study is different because it uses non-rigid objects, background objects and Deep Lab Architecture[3]. [17] use a voting method to detect parts, our study differs from theirs mostly by using non-rigid and background objects, and using a different unsupervised metric for the clustering algorithm

Chapter 3

Deep Neural Networks

Deep Learning takes in raw input image data and outputs a value for either the entire image or each pixel of the image representing the object it represents. Between the input layer and the final layers, there are multiple hidden layers. These hidden layers try to extract larger information from previous layers, in other words each layer tries to learn something about the image from previous layer. The way it does that is by giving features from previous input different weights. It assigns the weights by a mechanism called backpropagation, which carries information forward and backward between input and output layer, which ensures the right weights are assigned in hidden layer. To truly understand this it will be helpful to examine backpropagation in detail.

3.1 Backpropagation

Backward propagation of errors (Backpropagation) is a set of rules (algorithm) for guided learning of neural networks using a gradient descent [10]. The procedure computes the gradient of the functional error, keeping in focus the neural systems' weights. In other words, it calculates the effect of each neuron on the error function. It does so by the computation of gradient moving in reverse through the framework, with the slope of the last layer of weights being processed first and vice versa.

Backpropagation is a broad process, and it can be separated into four primary equations:

$$\delta_j^L = \frac{\partial C}{\partial a} \sigma'(z_j^l) \qquad \text{Equation 1}$$

$$\delta_j^L = ((w^{l+1})^T \delta^{L+1} \mathrm{N} \sigma'(z_j^l)) \qquad \text{Equation 2}$$

$$\frac{\partial C}{\partial b} = \delta \qquad \text{Equation 3}$$

$$\frac{\partial C}{\partial w} = a_k^{l-1} \delta_j^l \qquad \text{Equation 4}$$

Represents the error in the output layer, and j is the output neuron. The first term on the right hand side, δ_j^L, measures how much the cost function is affected by the jth output and the second term $\sigma'(z_j^l)$ measures how fast the activation function is changing.

The second equation calculates the error in one term due to the error in previous term. The W represents the weight matrix between the two layers, and the is the δ^{L+1} error in the next level. Hadamard product moves the error backward through the activation function giving us the error δ^L.

The third equation calculates the rate of change of the cost with respect to any bias in the network.

The fourth equation on the other hand calculates the rate of change of the cost function in terms of each neuron. One nice feature of that is that if the activation is small the learning rate is small and vice versa. This is because the study focuses on neurons that have the most effect, or in other words can have the biggest change in magnitude. Another side effect is that the network will become sparse as the neurons with value of zero stop learning.

Using the 3rd and 4th equation we can now update the weight of each neuron by multiplying their weights by the gradient descents calculated.

To summarize, during a training phase the Neural Networks, input data are pass forward from the input layer to the hidden layers and ultimately to the output layer. Then the output error is computed, then by using backpropagation we update weight of each neuron, and propagate the data through the network again and repeat. This workflow is usually repeated for thousands of times.

Chapter 4

Training Network

In order to fully train on all dataset we need to train our networks on background and foreground images using the ground truth. We used the PASCAL-CONTEXT[9] dataset images on 33 objects, 20 foreground and 13 background. After training, we take the prediction results from predefined interested layer and finally cluster them.

4.1 Dataset and Preparation

The PASCAL-CONTEXT dataset contains pixel-wise labeling for the 10,103 trainval images of the PASCAL VOC 2010 detection challenge. Figure 1 shows objects in actual image and the pixel-wise labeling in the second image. There are 540 different classes in the dataset, which can be further divided into three types: (i) objects, (ii) stuff and (iii) hybrids. The objects are defined as things with structured and well defined shape, this class includes the original PASCAL 20 objects. While stuff represents objects that do not have a well-defined shape such as sky, water. The hybrid are the type of the object that have boundaries but no predefined shape, an example would be road.

From the 540 categories we choose 33 categories that Mottaghi [9] showed we have enough data to be able to do optimal studies on them. The remaining 507 objects are classified as “others”. In addition PASCAL 20 objects (foreground), we picked the following 13 objects: building, ceiling, floor, grass, ground, keyboard, mountain, road, sky, track, tree, water, wall, which comprised the background/context data.

The pascal dataset comes with labeled training and val test. We use the training set in order to train the data and the Val set for the evaluation.

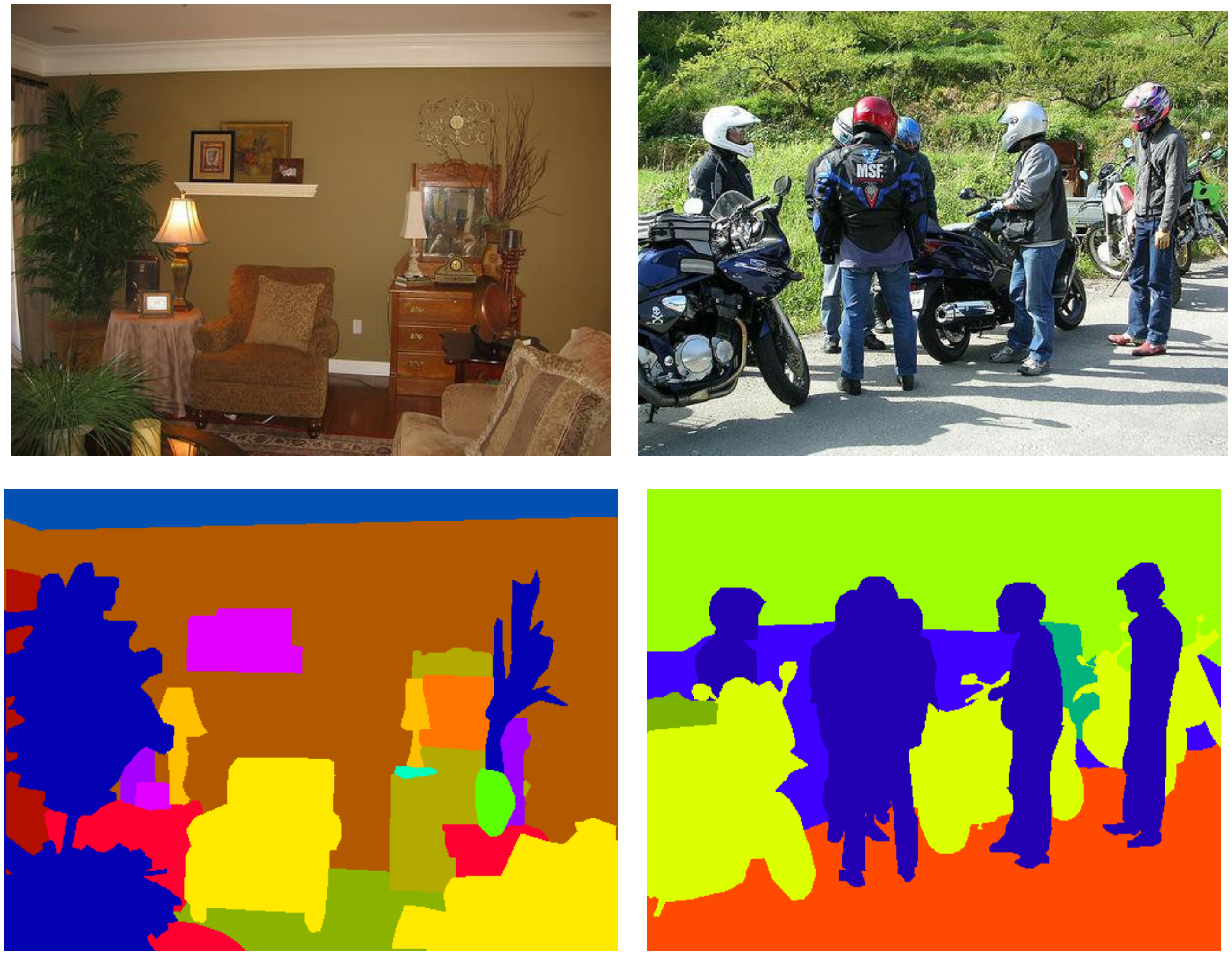


Figure 1 The images on top are the actual provided images in the Pascal dataset, on the bottom is the pixel wise labeling/ As you can see the dataset not only labeled the foreground objects, but the background objects such as road, grass and trees as well. Reference for images to Mottaghi

4.2 Network Architecture:

We used Deep-Lab, rather than the more popular VGG[11], as our network architecture. The reason being is that, we were able to achieve higher prediction power using Deep Lab-architecture than any other architecture. Secondly, VGG focuses on big features and bounding boxes, while Deep-Lab is focused on learning more detailed features, as its a pixel by pixel predictor, and as a result can capture textural details better than VGG can. This is important because many of the background objects, such as water and grass contain texture, where foreground objects like humans and cars do not.

4.3 Deep-Lab Architecture:

Introduced by Chen[3], the architecture has different layers, the first two layers are a convolution of 3x3 kernel, followed by RELU, followed by convolution of 3x3, followed by RELU, and max pooling of stride 2. The next two layers are almost identical to the first two layers, except that the 2nd RELU is followed by convolution of 3x3 and another RELU before a max pooling of stride 2. The 5th layer is identical to the 4th layer except that the convolution layers include a hole algorithm of size 2. The 6th and 7th layer is a fully connected layer, with hole kernel of 4, followed by RELU and a dropout ratio of size 0.5. The last layer is a convolution layer to the 34 objects/background output followed by a softmax layer.

We use the CAFFE library [6] for the training. We set the learning rate is set 0.001, then every 2,000 iterations reduce it by 0.1. The momentum and the weight decay are set to

0.9 and 0.0005 respectively. Finally a total of 6,000 iterations are done for the calculations.

In order to improve the performance of the Deep Lab algorithm, we used a series of Data Transformation. Following similar approaches to [3][5][8] cropping and normalization are performed. Each input image will have a pixel 36 as its small size, as later section show, this will ensure our images at least have a pixel ratio of 336. Secondly, we calculate the mean pixel value for all pixel over the entire PASCAL-CONTEXT dataset and subtract that from every image, to normalize the values.

4.4 Deep-lab vs VGG

Deep-lab architecture is based on VGG-16's with few differences:

1. The fully-connected layers are turned into convolutional ones.
2. Subsampling after the last two max-pooling layers are replaced by introducing zeros in convolutional filters in layers 'hole algorithm'
3. The layers are sparsely subsampled using input strides of 2 or 4 pixels.

VGG nets are known to provide too much invariance, a good thing for training networks for object classification, yet decreasing the performance for object segmentation due to the downsampling problem. In order to solve the problem, Deep Lab applies the "hole algorithm" (also known as "atrous algorithm").

Input (224 X 224 X 4)
Layer 1 Conv 3x3x64 RELU Conv 3x3X64 RELU Max Pooling 2
Layer 2 Conv 3x3X128 RELU Conv 3x3X128 RELU Max Pooling 2
Layer 3 Conv 3x3X256 RELU Conv 3x3X256 RELU Conv 3x3X256 RELU Max Pooling 2
Layer 4 Conv 3x3X512 RELU Conv 3x3X512 RELU Conv 3x3X512 RELU Max Pooling 2
Layer 5 Conv-Hole(2)3x3X512 RELU Conv-Hole(2) 3x3X512 Hole(2) RELU Conv-Hole(2) 3x3X512 RELU Max Pooling 2
Layer 6 FC-Hole-4096 RELU Dropout
Layer 7 FC-Hole-4096 RELU Dropout
Layer 8 FC-34
Soft-max

Figure 2 demonstrated the architecture of deep lab

4.4.1. 'Hole Algorithm'

Hole algorithm was originally created for making the computations of undecimated discrete wavelet transform much more efficient.

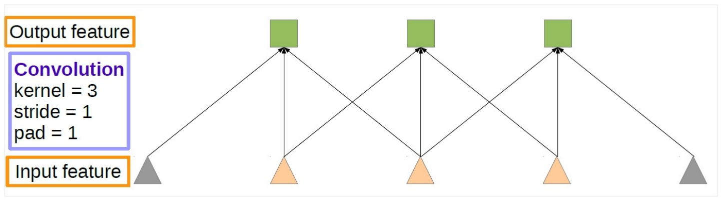
The algorithm is the following:

$$y^{[i]} = \sum_{k=1}^{K} x[i + r \cdot k] w[k] \qquad \text{Equation 5}$$

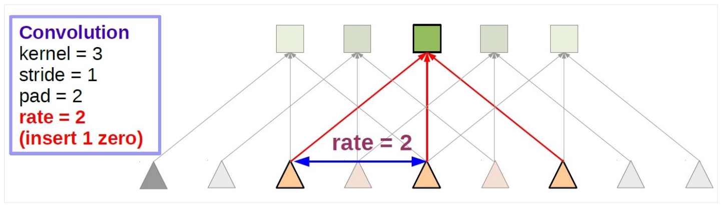
Where X is the signal w[k] is the weight of the neuron and r is the sampling rate. Fully convolutional network is a special case with sampling rate of 1. The r in the equation allows us to introduce 0 or 'hole' in the kernel.

4.4.2. Effect of 'Hole Algorithm'

The improvement occurs because traditional max pooling would select the weight of one of the two pixels, following that up by upsampling to a previous layer, only the value of 1 out 4 pixels would be selected. However, by using atrous method and introducing zeros and sampling from each filter, the networks will keep each node and resolution by introducing zero. Furthermore, the convolution layer with input values of 0, will have the same complexity as a smaller kernel, as the multiplication by 0 go to 0.



(a) Sparse feature extraction



(b) Dense feature extraction

Figure 3 top shows traditional convolution, below shows atrous convolution with rate =2. In the bottom, we see atrous convolution. In the top images the newly formed image from the kernel has a smaller resolution, while on the bottom the resolution is the same (same amount of green and orange boxes), yet the computation stays the same because the extra nodes have the value of 0. Reference for images to Chen

In other words in traditional VGG in order to reduce the calculation complexity, first we would apply max pooling(effectively selecting 1 out of 4 nodes), then do a fully connected convolution, then upsample to assign each pixel a value. With atrous, instead of max pooling, we introduce zeros and keep the same resolution by increasing the kernel size. But because the complexity of multiplying by 0 is 0, the complexity of the convolution is the same as the VGG, but this team each pixel contributes to the output instead of 1 out of 4 pixels after max pooling. Yielding to a more accurate result. Atrous method also allows us to artificially increase the field of view. For example the size of this network's receptive field in Deep Lab , through the use of spatial subsampling and introducing zeros, allows it to reach 128x128 and 308x308 receptive field sizes. This method the network optimize for localization of content while scanning a larger field of view.

To summarize atrous allows for three things to happen:

1. Keep the same resolution
2. Keep the same complexity
3. Increase field of view if needed.

Chapter 5

Pascal Context Dataset Results

5.1 Training and Results

We used the already predefined Train and Val images given by Pascal-Context to divide the dataset into training and val set. Instead of training Deep Lab on the original 20 objects, we modified the last layer to content 33 objects: the original 20 objects plus the 13 background objects.

We test the performance of Deep Lab on PASCAL-CONTEXT dataset by feeding it the valuation set, giving us a pixel by pixel prediction. Then we do a pixel by pixel comparison between the output and provided labeling for the training set. We Followed that by running an IOU calculation. We were able to reach an IOU score of 39.981%, while the CRF Berkley, FCN-8 group reached a score of 35.1%[8], and CRF-RNN had a score of 39.28% Zheng[20]. At the time of the study this was the highest score to be achieved. Table 1 shows the scores of all 59 objects, for compared to running it on 20 objects.

As Table 1 shows adding context to the data improves the IOU accuracy of the original data. On Pascal 2010 imageset, the accuracy of the original 20 objects improves from 57.17% to 57.92%. This is mainly to the networks ability to differentiate between similar looking objects and different objects that gives it context. This highlights the importance of understanding and training on context data.

	Pascal content	**Pascal original**	**Differenc e**
Backgroun d	25.86%	86.55%	N/A
Aero plane	64.38%	62.94%	-1
Bicycle	57.33%	57.45%	0
Bird	67.06%	67.24%	0
Boat	47.83%	44.72%	-3
Bottle	52.70%	52.93%	0
Bus	72.43%	72.73%	0
Car	64.38%	63.27%	-1
Cat	73.97%	75.47%	1
Chair	27.81%	25.18%	-2
Cow	63.64%	58.02%	-5
Dining table	34.51%	29.20%	-5
Dog	69.67%	70.73%	1
Horse	62.97%	60.33%	-2
Motorbike	66.99%	62.03%	-4
Person	74.77%	75.73%	0
Potterplant	46.61%	43.14%	-3
Sheep	67.19%	68.19%	1
Sofa	39.78%	37.83%	-1
Train	67.01%	65.40%	-1
TV monitor	52.25%	50.99%	-1
Bag	0.04%	N/A	N/A
Bed	2.12%	N/A	N/A
Bench	0.00%	N/A	N/A
Book	21.00%	N/A	N/A
Building	44.77%	N/A	N/A
Cabinet	21.20%	N/A	N/A
Ceiling	37.01%	N/A	N/A
Cloth	3.45%	N/A	N/A
Computer	3.40%	N/A	N/A
Cup	14.43%	N/A	N/A
Door	5.73%	N/A	N/A
Fence	25.05%	N/A	N/A
Floor	45.94%	N/A	N/A
Flower	16.16%	N/A	N/A

Food	44.95%	N/A	N/A
Grass	73.20%	N/A	N/A
Ground	45.33%	N/A	N/A
Keyboard	40.37%	N/A	N/A
Light	7.56%	N/A	N/A
Mountain	42.22%	N/A	N/A
Mouse	0.00%	N/A	N/A
Curtain	28.45%	N/A	N/A
Platform	30.84%	N/A	N/A
Sign	23.34%	N/A	N/A
Plate	24.40%	N/A	N/A
Road	42.74%	N/A	N/A
Rock	27.76%	N/A	N/A
Shelves	13.03%	N/A	N/A
Sidewalk	9.04%	N/A	N/A
Sky	88.15%	N/A	N/A
Snow	49.49%	N/A	N/A
Bedclothe s	26.63%	N/A	N/A
Track	46.18%	N/A	N/A
Tree	66.76%	N/A	N/A
Truck	5.33%	N/A	N/A
Wall	50.15%	N/A	N/A
Water	75.68%	N/A	N/A
Window	24.82%	N/A	N/A
Wood	8.23%	N/A	N/A
IOU	38.98%	58.58%	

Table 1 Shows the IOU score of the 59 objects Pasca dataset. The first column is the object, the second column is the Pascal-context IOu score, the third column is the Original Pascal 20 objects, the 4th column shows the difference between the original 20 objects. As you could see most object had a worst performance without the context.

Chapter 6

Clustering:

We cluster the activation neurons of an image corresponding to different parts of an object at a given layer the network. Given the sparsity of the data we use the K-median algorithm to cluster the data, from this we will create a dictionary of parts that we will use in Chapter 7.

6.1 Image Selection:

For training we use the training set of pascal context dataset. Each image is contained of multiple objects, and some objects appear more than once in an image. For example, in Figure 1 multiple humans and motorcycles are shown. Given that we have the pixel by pixel labeling and index labeling of each object, we can create a bounding box around each object. Therefore for each image we do the following:

1. Take the minimum and maximum position of each object instance, in both the width and height axis. When the labeling has a discontinuity we treat that as 2 separate objects.
2. Create a bounding box, around each instance.
3. Re-size each bounding box to Deep-lab input size of 224 pixels. This is done by making the shortest side 224, using average extrapolation of neighboring pixels

4. Then subtract the deep lab average pixel color for each pixel of the newly formed image, and normalize it.
5. Then feed the image forward to the network.

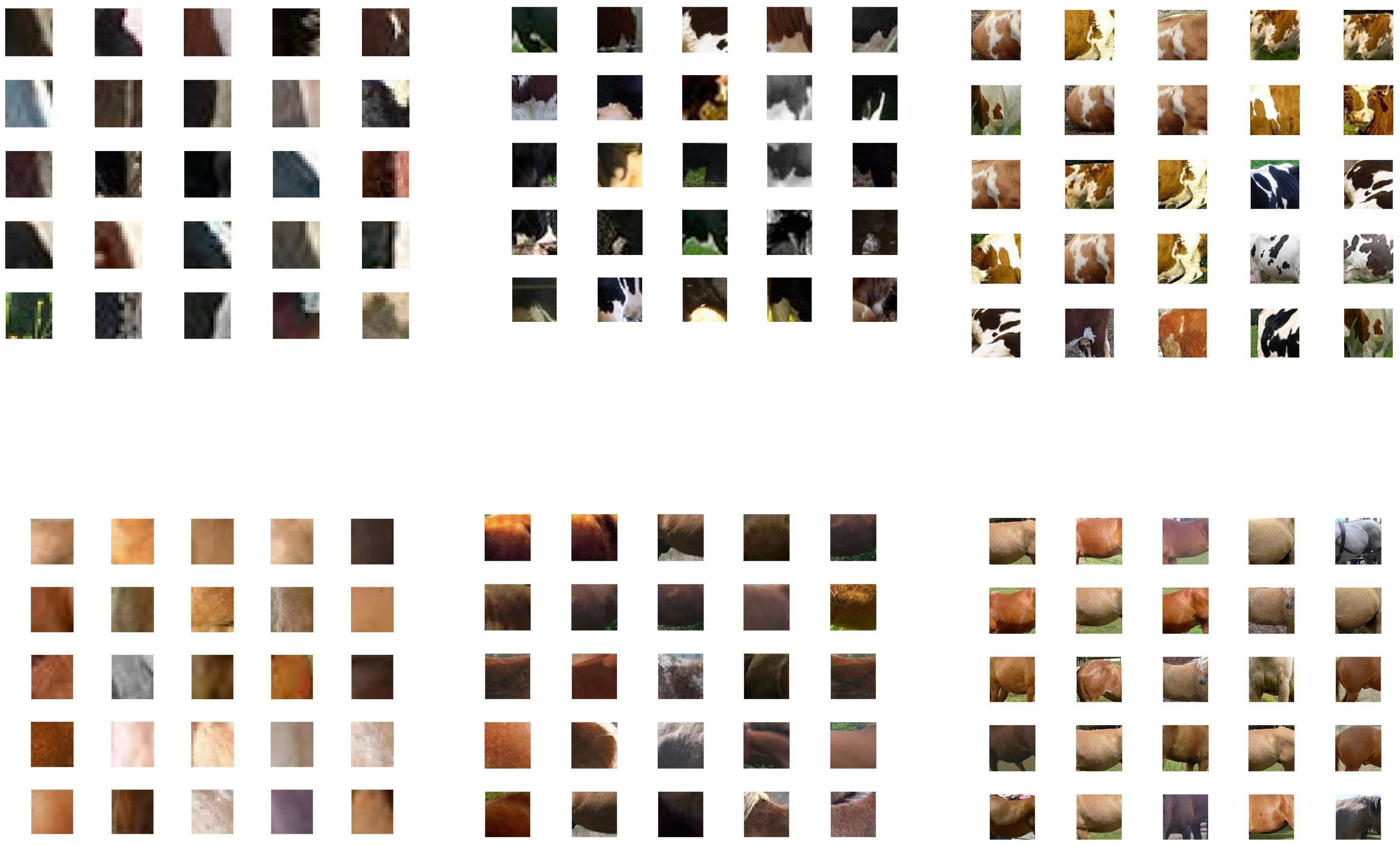


Figure 4 demonstrates the patch level detail that the neuron studies, the left column is at layer 2, the center column is layer 3, and third column layer 4. Top images 3.1 a, shows the output for cow, while 3.1b shows the output for a. As you can see layer 2 is good for capturing texture, but not large enough to predict parts, layer 3 allows us to detect parts, and layer 4 allows us to detect object as well.

6.2 Layer Feed Network:

For each renown image we feed it into the trained Deep-Lab network and pass it forward through the network. We then forward into the the 4th layer and store the value at each neuron.

The reason that we use the 4th layer is due to the corresponding image sizes in that layer are detailed enough to capture the textures and large enough to be visible to human eyes. Figure 4 shows the difference between the different layers.

The way the mapping works in the 4th of the layer in Deep-Lab Architecture is similar to VGG-16, for the first 4 Layers. Mainly, as Wang shows, given image has x B $R^{W\times H\times 3}$, which yields to a response mapping of $f`_l$ B $R^{Wl\times Hl\times Nl}$ where the l corresponds to the layer of the deep-network, and N to the number of neurons in the layer.

Then from the image we randomly sample and select 100 points. We randomize to by having the computer select 100 different points in grid format, creating a patch of $P^{i,j}$ BR^{N} . We take the neuron reaction for this 100 point and store them. Then we calculated the padding and stride effect on an image, and using reverse upsampling we find the bounding box of the portion of the image corresponding to that neuron.

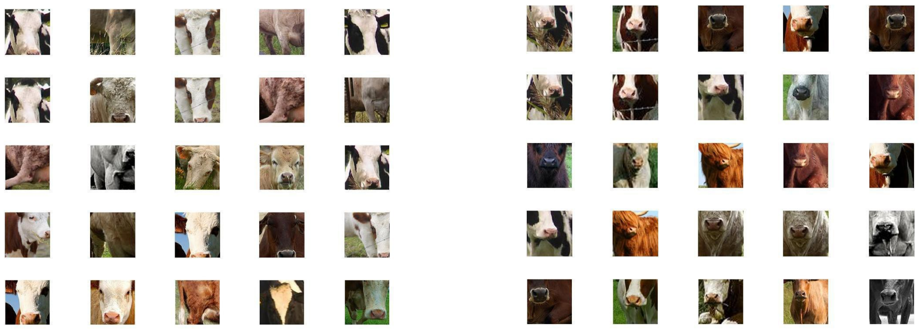


Figure 5 The image on the left(4.2a) corresponds to a cluster formed on faces of cows, while the cluster on the right(4.2b) captures neurons corresponding to the nose of the cow.

6.3 Clustering:

Given that the neuron corresponds to a small portion of the image, many times multiple sets of neurons correspond to the same part of an image. For example, as Figure 4 shows, for cow it we can have multiple sets of neurons corresponding to the image of the face of cows from different angles. Therefore, we need to construct a way for grouping the neurons together in an unsupervised fashion. The traditional k-means algorithm is usually used for this types of tasks.

6.4 K-means:

The k-Means clustering problem in 1D is defined as follows. Given X = {x1, ..., xn} ъ R and k, find centroids M = {µ1, ..., µk} ъ R minimizing the cost

$$\sum_{x\,a\,X} min \quad {}^{\wedge}\mathrm{M}\,(\mathrm{x}\ -\quad)^2 \qquad \text{Equation 6}$$

This is a NP-Hard problem forcing people to use heuristics approaches, and even then it’s not an easy task. [4].

For default we use the Lloyd’s algorithm: iterative local search heuristic that has a running time of O(tkn) where t is the number of rounds of the local search procedure. To further improve that we combined it with kmeans++ as proposed by Arthur, that can speed up and optimize Lloyd's method.

In theory, if Lloyd’s algorithm is run to convergence to a local minimum, t could be exponential and there is no guarantee on how well the solution found approximates the optimal solution [4]. Lloyd’s algorithm is often combined with the effective seeding technique for selecting initial centroids due to [2] that gives an expected O(lg k) approximation ratio for the initial clustering, which can then improved further by Lloyd's algorithm.

6.5 K-Median:

Is similar to the K-mean algorithm. The only difference is that instead of using Euclidean distance, it will use the Manhattan distance.The K-Medians problems is also NP-hard and the best polynomial time approximation algorithms has an approximation factor of 2.633 [4].

6.6 Advantage of using K-Media Over K-mean:

The activation of neurons in layer 4 has a sparsity of 68.9% on average, while k-means calculates the average of each neurons, and out of thousand of images one non-zero corresponds for a neurons will lead to a non-sparse value on the aggregate level, as a results on average 99% of the data are non-sparse.

K-medians on the other hand keeps the sparsity level as long as more than half of the neurons do not get activates. Hence the responding results will have a sparsity level of 37%. Consequently we saw an improvement of 7% in accuracy on the subset of interested background objects.

6.7 Number of Clusters:

The numbers of clusters for each object differed significantly, in order to choose the optimal number of clusters, for each object we plotted the elbow plot, the method introduced by Thordnike[13].

However, this number of clusters was still not optimal, as both outliers and “others” would form their own clusters. In order to remove them in an automatic form, we removed images belonging to a cluster based on three conditions:

1. If the cluster size was smaller than 50 patches, as it means that the cluster is usually cluster of outliers.
2. If the distance to other clusters was larger than twice than the median of distances between clusters; as it implies the clusters are an obsolete objects.
3. If the variance is more than twice the average variance within clusters, remove them, this indicates that the cluster is not tight and does not represent a part of an object.

Chapter 7

Evaluation

For each object a dictionary of parts is comprised of the different clusters created through the K-median step. A dictionary of dictionaries is created when all these clusters, from different clusters are combined together into one cluster.

When a new formed bounding box from the validation set are given to Deep-Lab, the values at the layer of interest are extracted. Then the Manhattan distance is used on the corresponding set of activated neurons and all the clusters to locate the nearest cluster. That nearest cluster is what our algorithm believes is the true cluster that the object belongs to and marks it as our prediction.

7.1 Prediction step:

For each object a dictionary of parts is comprised of the different clusters created through the k-median step. A dictionary of dictionaries is created when all these clusters are combined together into one cluster.

When a new formed bounding box from the val images is given to Deep-Lab and feed forwarded to the interested layer, we can take activate neuron of that layer. Then we can use the Manhattan distance on the corresponding set of activated neurons and all the clusters to locate the nearest cluster. That nearest cluster is what our algorithm believes is the true cluster that the object belongs to and marks it as our prediction.

7.2 Splitting the data

The pascal dataset is already divided between train and valuation set, we will only use the train set to create our dictionary of dictionaries while we will use the valuation set to predict the accuracy of our algorithm.

7.3 The Challenge with IOU:

The images that are feed forward through network are created through a bounding box, followed by random sampling, which leads to a smaller bounding boxes. This small bounding box could compromise from multiple objects as shown in Figure 6, a patch could contain both a horse and a grass. However, our algorithm is limited to predicting one cluster that belongs to one object. Thus our algorithm could only predict one object, even though many objects could be present in the bounding box. Therefore, the IOU is not ideal for this problem because:

1. The accuracy and precision will be deflated as the true answer could be a combination of objects, but the system is forced to predict one, and thus getting a penalty for the wrong answer. For example, if we have a bounding box of horse, if the object predicts grass, even though half the image is grass, it will get penalized for not predicting horse.
2. We will penalized the system for correctly predicting the more sophisticated foreground object than the background. As demonstrated in Figure 6

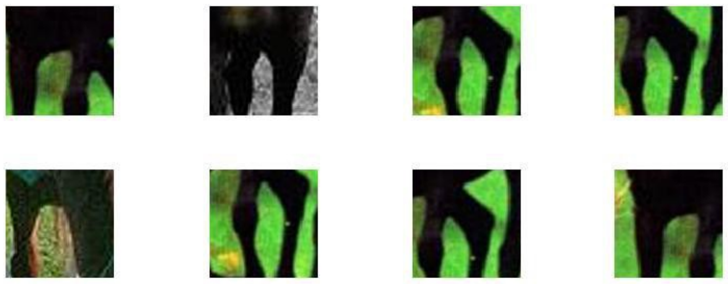


Figure 6 are a set of images that show an image of a horse and a background, even though the leg is the object that stands out the most, the grass occupies more pixels than the horse

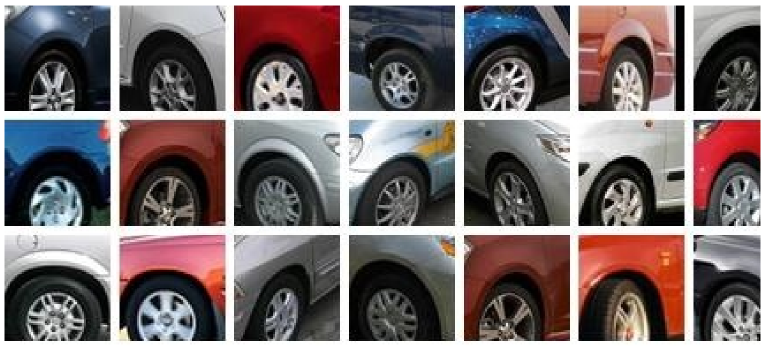
7.4 Our Metric:

Given the difficulty of the task we created a new metric. We will reward our prediction 1 point if any of the following criteria is made and 0 points if it does not:

1. If over 67% of the bounding box is from a foreground object and the foreground object is predicted
2. If over 67% of the bounding box is from a background object and either i) the background object is chosen or 2) the most occurring foreground object is chosen
3. If no image composes of 67% then if the most occurring foreground object or one of the top 2 occurring background object is met a point is given.

The advantage of our metric is that it allows for multiple correct answer when there are multiple possible answers, and it allows for only one truth value when one truth value arises. Compare to IOU, where it only allowed for one prediction when multiple possible answers relies. However, this method allows for false positive as well.-

Using our metric we can see an average F1 score of 22.1%. Ideally we will have a dataset that has both the whole objects and its parts labeled. then we can check the accuracy by seeing which part is predicted, which will give us a more accurate metric to evaluate the prediction.



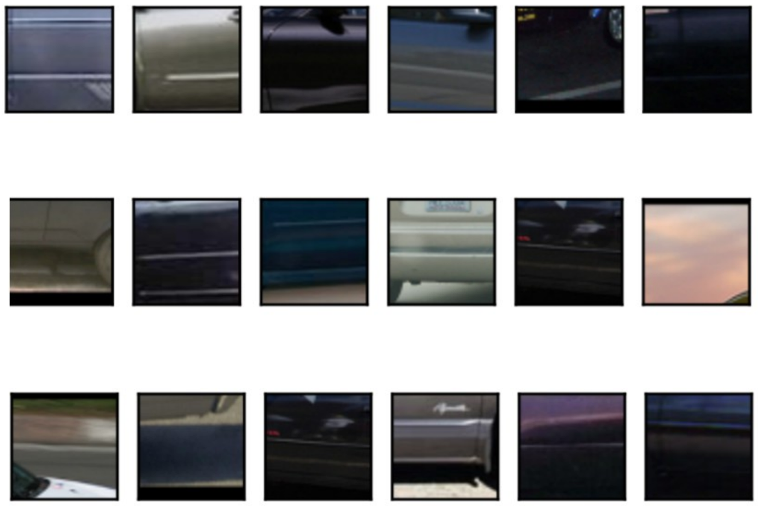


Figure 7 Top are the car patches created by Wang, they focus on larger parts of objects. Bottom are the car patches created by our method, the patches focuses on texture and granular details, such as lines being parallel.

7.5 Results and Discussion:

Our results and contribution is both quantitative and qualitative.

Qualitative:

Above in Figure 7 we have showed the clusters for car created by our method and compared to Visual Cluster work of Wang. As you can see the results are slightly different. The clusters produced by Wang were tight around the larger patterns in the image, while the results of our work focused more on the texture of the object and the background. This is precisely because of the underlying architecture of the network, discussed in Chapter 4 Wang used VGG-16 and we used the Deep-Lab architecture. The VGG architecture was trained on Imagenet Challenge, which is focused on object detection and focuses on larger patterns. While Deep-Lab was trained on pascal and was more focused on pixel-by-pixel training. Consequently, our clusters have a higher granularity in their learning, and a stronger ability to learn background objects at the cost of visualization not looking as close as the VGG architecture.

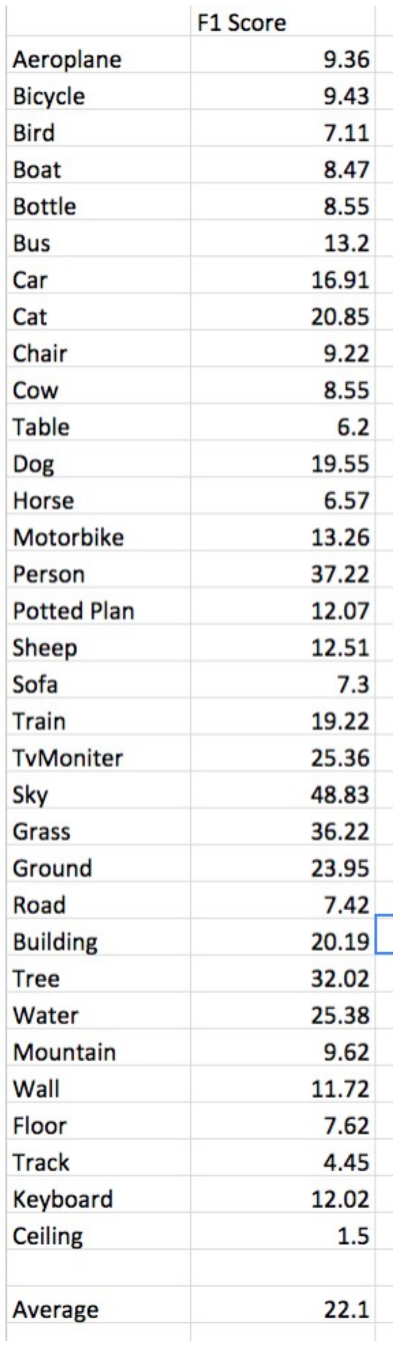
Our second difference is by the amount of clusters. The deep-lab architecture focuses on more granular texture, most background objects have similar type of texture therefore, the more clusters we have we noticed less meaningful the additional clusters became. Consequently ,the amount of clusters needed to capture the generalize shape of clusters were smaller. For example , cars had 200 clusters before puning and 147 after punning in Wang’s[17] work, while in our architecture they had 15. The advantage of that is that the search space of the prediction algorithm is smaller than the visual cluster of wang, which by default increases accuracy.

Quantitative:

Results of the different objects are summarized in Table 2. Given that this is a multi class unsupervised network and we can only have 1 prediction, when multiple objects exists the accuracy will be low. We looked at the F1 score of our metric and a score of 22.1% was observed.

For most of the objects the precision score was lower than the recall decision. One reason, is that using our metric the algorithm has to decide between one of the highest confidence interval. It already has eliminated many of its choices. The deductive algorithm has limited the search space.

Furthermore this accuracy was achieved because using our approach each object has about a quarter of amount of clusters representing it. These reduces the search space and thus increases the probability of guessing correctly.



	F1 Score
Aeroplane	9.36
Bicycle	9.43
Bird	7.11
Boat	8.47
Bottle	8.55
Bus	13.2
Car	16.91
Cat	20.85
Chair	9.22
Cow	8.55
Table	6.2
Dog	19.55
Horse	6.57
Motorbike	13.26
Person	37.22
Potted Plan	12.07
Sheep	12.51
Sofa	7.3
Train	19.22
TvMoniter	25.36
Sky	48.83
Grass	36.22
Ground	23.95
Road	7.42
Building	20.19
Tree	32.02
Water	25.38
Mountain	9.62
Wall	11.72
Floor	7.62
Track	4.45
Keyboard	12.02
Ceiling	1.5
Average	22.1

Table 2 shows the F1 scores using our metric as an evaluation. We can see that the accuracy is lower than supervised learning, but higher than coin-flip. Also, notice that bigger objects tend to do better.

CHAPTER 8

Conclusions and Future Work

In this paper, we extended deeplab segmentation work from the original 20 objects to 33 objects, and by doing so it enabled us to present an unsupervised method of creating visual clusters and dictionary of parts from supervised networks. Even though such ideas have been explored in the past we contributed and improved the results in four main ways: we extended from a small set of objects to 33 objects and doing so achieved the highest accuracy on the pascal context dataset, secondly we improved the clustering by using k-median instead of k-means, thirdly our cluster captures more texture and granular details than Wang's work and finally we introducing a new correction metric.

Our method has two shortcomings one is due to data and another due to the nature of clustering. The first is that we do not have a rich dataset with both objects and parts labeled, currently our algorithm gets a free pass if it predicts the right object but the wrong cluster. By having parts labeled we can instead measure if the correct part has been labelled. Secondly, we only allow for one prediction in future, we should change the prediction method to allow a belief of strength for each cluster, and if the score is above a certain value for it to be among the correct answers.

References

[1]] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In ECCV. Springer, 2014.

[2] Arthur, D. & Vassilvitskii, S. K-means: The Advantages of Careful Seeding. 2006

[3] Chen, L., Papandreou, G., Kokkinos, I. Murphy, K., & Yuille, A. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. 2015

[4] Grønlund, A., Larsen, K., Mathiasen, A., Nielsen, J., Schneider, S., & Song, M. Fast Exact k-Means, k-Medians and Bregman Divergence Clustering in 1D. 2017

[5] Krizhevsky, A., Sutskever,I., & Hinton, G., ImageNet Classification with Deep Convolutional Neural Networks. 2012

[6] Jia, Y., Shelhamer, E., Donahue, J., Karayev,S., Long,J., Girshick,R., Guadarrama,S., & Darrell,T. "CAFFE: Convolutional Architecture for Fast Feature Embedding." ACM International Conference on Multimedia, 2014.

[7] Li,Y., Liu, L., Shen, C., and Van den Hengel, A.. Midlevel deep pattern mining. In CVPR. IEEE, 2015.

[8] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In IEEE CVPR, 2015.

[9] Mottaghi,R., Chen, X., Liu,X., Cho,N., Lee,S., Fidler,S., Urtasun,R. & Yuille, A. The Role of Context for Object Detection and Semantic Segmentation in the Wild. 2014

[10] MuÌller, Berndt, et al. Neural Networks: an Introduction. Springer, 1995.

[11] Simonyan, K. & Zisserman, A. Very Deep Convolutional Networks for Large_Scale Image Recognition. 2014

[12] M. Simon and E. Rodner. Neural activation constellations: Unsupervised part model discovery with convolutional networks. In ICCV. IEEE, 2015.

[13] Thorndike, L. , M. Who Belongs in the Family. 1953

[14] Wang, J., Lin, X., Zhang, Y., Lee, T., & Yuille, A. Learning Robust Object Recognition Using Composed Scenes from Generative Models. 2017.

[15] Wang, J., Xie, C., Zhang, Z. Zhu, J., Xie L., & Yullie, A. Detecting Semantic Parts on Partially Occluded Objects. 2017.

[16] Wang, J., & Yullie, A. Semantic Part Segmentation using Compositional Model combining Shape and Appearance. 2014.

[17] Wang, J., Zhang, Z., Xie, C., Premachandran, V., & Yuille, A. Unsupervised learning of object semantic parts from internal states of CNNs by encoding. 2015.

[18] Wang, J., Zhang, Z., Xie, C., Zhou, Y., Premachandran, V., Zhu, J., Xie L., & Yullie,A. Visual Concepts and Compositional Voting. 2017.

[19] XiaoT., Xu, Y., Yang, K., Zhang,J., Peng,J., & Zhang, Z. The application of two-level attention models in deep convolutional neural network for finegrained image classification. In CVPR. IEEE, 2015.

[20] Zheng,S., Jayasumana, J., Romera-Paredes, B., Vineet,V., Su,Z., Du,D., Huang,C., & Torr,P. Conditional Random Fields as Recurrent Neural Networks. 2016.