ECE 1512: Digital Image Processing and Applications

UNIVERSITY OF TORONTO

Course Number: ECE 1512
Date Due: 02/12/19
Date Handed in:02/12/19

Final Project Report

Group members:

Sr No	Student Name	Student Number
1	Sam Sattarzadeh	1003060581
2	Yogesh Iyer	1005603438

University of Toronto

ECE1512
Group Final Report

Title: **Localizing Vehicles via Semantic Segmentation**

Project I.D.#:	1F	
Team members: (Select one member to be the main contact. Mark with ‘*’)	Name:	Email:
	Sam Sattarzadeh *	Sam.sattarzadeh@mail.utoronto.ca
	Yogesh Iyer	yogesh.iyer@mail.utoronto.ca
Supervisor:	Prof Kostas Plaitiniotis	
Submission Date:	Dec 2nd 2020	
Additional Comments		

Group Final Report Attribution Table

This table should be filled out to accurately reflect who contributed to each section of the report and what they contributed. Provide a **column** for each student, a **row** for each major section of the report, and the appropriate codes (e.g. ‘RD, MR’) in each of the necessary **cells** in the table. You may expand the table, inserting rows as needed, but you should not require more than two pages. The original completed and signed form must be included in the hardcopies of the final report. Please make a copy of it for your own reference.

Section	**Student Names**			
	Sam Sattarzadeh	Yogesh Iyer		
Problem Statement and Motivation	RS, RD	MR, ET		
Background Theory	RS, RD	MR, ET		
Result Analysis	RS, RD	MR, ET		
Conclusion	RS, RD	MR, ET		
All	FP	CM		

Abbreviation Codes:

Fill in abbreviations for roles for each of the required content elements. You do not have to fill in every cell. The “**All**” row refers to the complete report and should indicate who was responsible for the final compilation and final read through of the completed document.

RS – responsible for research of information
RD – wrote the first draft
MR – responsible for major revision
ET – edited for grammar, spelling, and expression
OR – other
“All” row abbreviations:
FP – final read through of complete document for flow and consistency
CM – responsible for compiling the elements into the complete document
OR - other
If you put OR (other) in a cell please put it in as OR1, OR2, etc. Explain briefly below the role referred to:
OR1: enter brief description here
OR2: enter brief description here

Signatures

By signing below, you verify that you have read the attribution table and agree that it accurately reflects your contribution to this document.

Name YOGESH IYER **Signature** **Date: 02-DEC-2019**

Name SAM SATTARZADEH **Signature** **Date:02-DEC-2019**

Name **Signature** **Date:**

Name **Signature** **Date:**

Contents

CHAPTER 1: Problem Statement & Motivation

1.1 Weakly-Supervised Semantic Segmentation: An Introduction

Semantic segmentation, is a novel image annotating approach, coming towards “Classification” and “Object Detection”. As we know, classification is the process of determining the type of object to which the input image belongs to, also called as “image-level label”; and “Object Detection” a process providing information about the location and size of the image. Semantic Segmentation deals with labelling individual pixels of the images as an instance of object classes.

Semantic segmentation is able to provide the consumers with more precise and detailed information about the existence and localization of multiple objects in comparison with classification and detection methods, making them functional in several applications such as medical diagnosis, robotic navigation, and scene recognition. However, implementing fully-supervised semantic segmentation models is too challenging and costly, since creating accurate annotation masks for large common data sets is a significantly time-consuming task which requires considerable investments. To overcome this issue, weakly-supervised semantic segmentation methods has gained interest in the recent years.

Weakly-supervised semantic segmentation approaches are aimed to discover a labelling model for the pixels of the images, based on initial and incomplete seed cues, or image-level labels of the train images. Throughout expanding these initial images to the whole image, these approaches can skip the exhausting process of strongly-labelling the train images, while obtaining a high-level understanding of the images and scenes.

1.2 Project Aim

For the scope of this project, we consider a semantic segmentation algorithm on the urban scenes, to investigate its performance on one of its most crucial applications, autonomous driving. Recent developments in image processing, especially object detection, put the researchers in this field towards inventing reliable smart self-driving cars. Since such cars are going to be designed for moving around streets, alleys, highways, and other parts of cities and megacities, it is vital for the groups working on implementing these smart vehicles to provide their autonomous cars with strong understanding of the urban scenes like traffic lights, road signs, pedestrians, and different types of vehicles; because even one false prediction from the side of the automotive cars might cause irrecoverable financial and health consequences.

For the sake of this project, the semantic segmentation network proposed by Huang et al. [1] will be described in detail, and its implementation and performance in recognizing urban objects and scenes will be analyzed. This method, a.k.a. Deep Seeded Region Growing (DSRG), is a deep neural network whose objective is to be trained for reaching initial cues of the image-level labelled data, and expand them in the order that the whole image is classified in either an object group or background. This algorithm makes progress in the previous Seeded Region Growing (SRG) [2] method which is utilized to pair all pixels of the images with some initial cues to categorize the image, based on some defined criteria. The main development made in SRG is that the proposed DSRG approach uses high-level semantic features extracted from the images instead of the basic ones such as color, intensity, and their gradients which were used in the former method.

1.3 Organization of report

The remainder of this report has been divided into three further chapters as follows:

Chapter 2: In this chapter, the background of DSRG is expounded. The process of classifying and segmenting images by this network is described in full details.

Chapter 3: In this chapter, the results implementing DSRG on urban scenes will be provided, alongside an analytic discussion about the advantages and drawbacks of the algorithm.

Chapter 4: The future plans to enhance the network in autonomous driving is pointed out.

It highlights the conclusions from the project and also the challenges faced.

CHAPTER 2: Background Theory

2.1. Introduction

In 2015, B.Zhou et al. explained the psychology of the classification process operated by neural networks, showing that convolutional units of these networks act like object detectors and scene descriptors, even they are not received any information about the location of objects during training phase. [3] According to this work, this remarkable phenomenon is ignored and discarded in many state-of-the-art classifiers. Later, they proposed Class Activation Map (CAM) method to reveal that classifiers are able to reach fruitful information about the location of different objects in their convolutional units [4]. In the other words, CAM is an Explainable Artificial Intelligence (XAI) algorithm pointing out the most involved features in predictions, based on the input image.

The main idea proposed in [4] was to take the localization ability of the networks back into the account, and using the introduced CAMs as both a seed cue generator, and the input features for a seed region growing algorithm. Integrating CAM and SRG, Huang et al. became able to collect and fine-tune labelling data cheaply and fast for all data sets, restricting the need of providing ground truth masks for multiple segmentation data sets to their evaluation set.

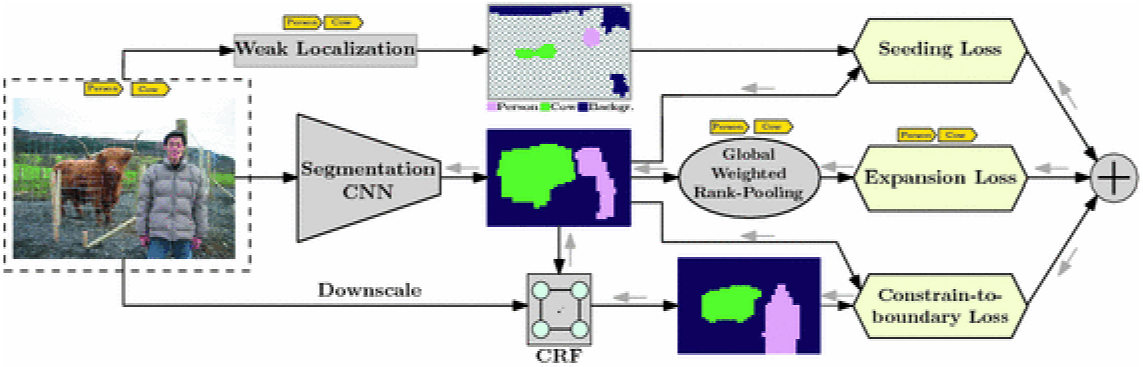


Figure 2.1: The flowchart of Seed, Expand, and Constraint (SEC) algorithm proposed by Kolesnikov and Lampert. [5]

2.2 Related Works

Before operating semantic segmentation in a weakly-supervised manner raised attention and interest, some techniques like discriminative classifiers [6, 7] and probabilistic graphical models [8, 9] had been introduced in the field of Machine Learning and Computer Vision. Getting deep into weak supervision, the further works aimed segmenting images for which a partial segmentation or a bounding box would be developed [10, 11]. The recent approaches for semantic image segmentation which focus on segmenting an image via training on image-level labelled data sets, are mainly use at least one of these contributions: (1) finding the cases of similarity between the features of test set and those of training set [12], (2) replacing the common loss functions with per-image ones in order to derive spatial representations of an image for maintaining the pixel-level class labels [13], and (3) self-training images according to pixel-level annotations derived from the body of the model.

A. Kolesnikov and H. Lampert designed an algorithm based on the latter two idea mentioned in the above paragraph [5]. Their methodology whose flowchart is summarized in Figure 2.1, provided a background for Huang et al. [1] to build their own DSRG segmentation algorithm.

The key idea between SEC [5] and DSRG algorithms are same. Both used the idea of self-training of a pre-trained network and defining a loss function appropriate to their segmentation approach Also, they both took CAM [4] into utilization to maintain some initial cues. But, Huang et al. [1] made an approach to generate more accurate seeds. They also used a different loss function to optimize their network. Figure 2.2 depicts the overview of DSRG algorithm. The rest of this section concentrates on explaining the details of DSRG methodology in more details, covering the seed generation process and the loss function used in this work, as well as brief comparisons between their approach and the previous work SEC [5].

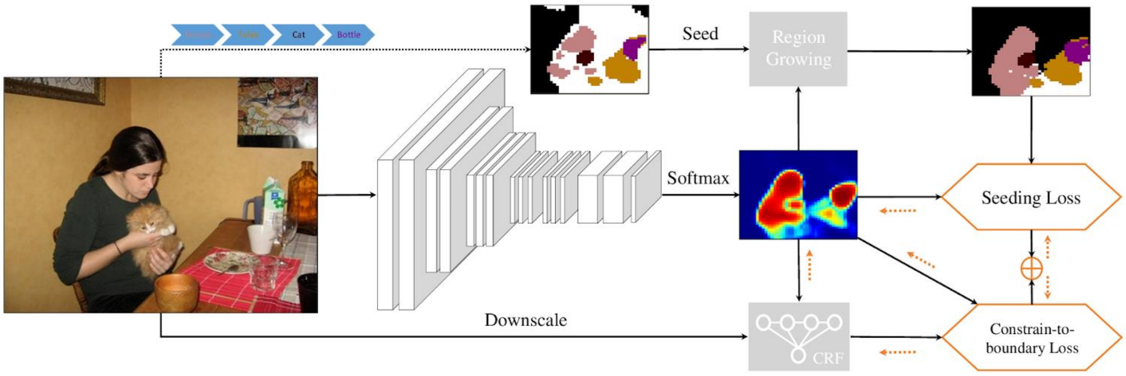


Figure 2.2: The flowchart of the network proposed by Huang et al. [1]

2.3. Class Activation map (CAM)

It is known that the deeper layers in each Convolutional Neural Network (CNN) reveal information about the existence of higher-level features in multiple spatial areas of images. While the first convolutional layers in CNNs are responsible for detecting simple features such as edges, corners, textures, and other kinds of regular patterns, the latter ones are adapted to browse for more complicated shapes, objects, and so on. This sensational ability of neural networks makes them functional in several applications like part-based object recognition. As mentioned previously, Class Activation Maps are implemented to recover these precious phenomena from the body of CNNs.

Zhou et al. [4] designed a modification which outputs activation maps for all classes from the feature maps located in the last convolutional layer. In their work, the fully-connected of a chosen network is replaced with a Global Average Pooling (GAP) layer, followed by a dense output layer. After training the modified network, for given test image, activation maps of each class is calculated by summation of the multiplication of each feature map with the weighting factors connecting their corresponding GAP node and the output node referring to the output class. It should be noticed that softmax layers which are commonly following the output layers are ignored in reaching CAMs due to the fact that they do have ignorable impact on the procedure of class prediction.

In the mathematical words, assuming $f_k(x, y)$ to indicate the feature map of the k-th node in the very last convolutional layer of a network at a given spatial location(x, y), GAP layer acts like an averaging function F^k as follows:

$$F^k = \frac{1}{Z}\sum_{x,y} \; f_k\,(x, y) \qquad (1)$$

In the above equation, Z represents the size of an image. Let the values of the weighting matrix linking the nodes of GAP layer and output fully-connected layer to be indicated with w_c^k, where c points out to an output class, for each output class c, Class Activation Map is maintained throughout equation (2) which is mentioned below:

$$M_c(x, y) = \sum_{x,y} w_c^k \ f_k\,(x, y) \qquad (2)$$

The values of $M_c(x, y)$ point out the more important spatial regions for activating an output node for a specific input image. In the main work, CAMs are used as a guideline highlighting the initial seed cues, which are going to be expanded based on the criteria which are constituted in SRG algorithm. The summary of the procedure of calculating activation maps is exemplified on providing an explanation for an Australian terrier in Figure 2.3.

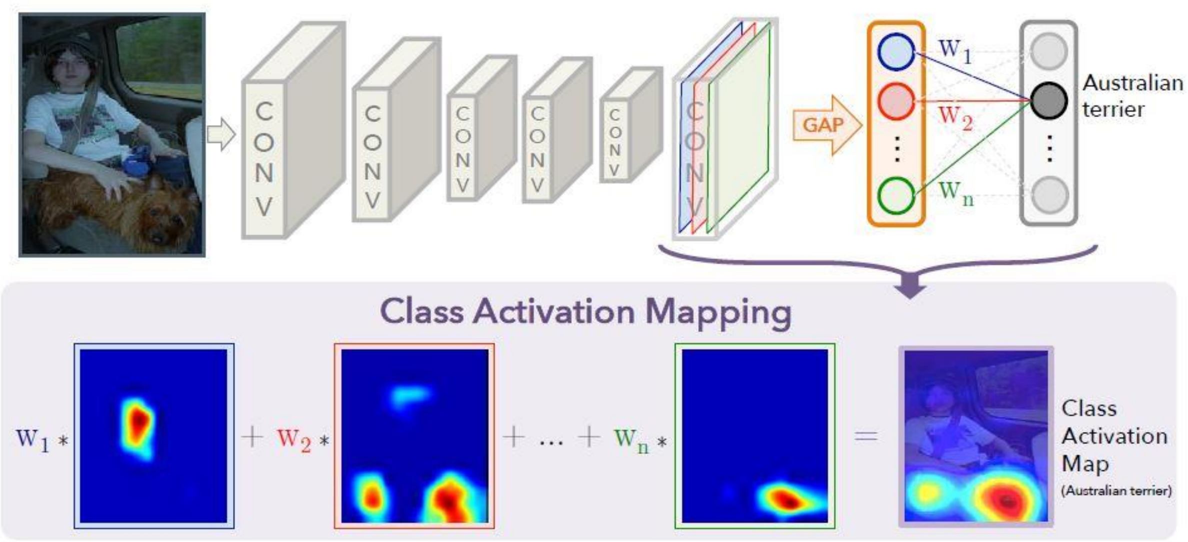


Figure 2.3: Getting a Class Activation Map from a Convolutional Neural Network.[4]

2.4. Background Extraction

Besides the object classes, this work uses a separate machine learning model to compute activation maps for background. Instead of assigning an output class named as "background" to the final layer of the base network, a saliency map is maintained, which is equivalent to the negative picture of a "background activation map". There are numerous different models have been presented for solving the problem of salient object detection. In the main method, a Discriminative Regional Feature Integration (DRFI) is used to extract background. As can be guessed from the name of this algorithm, in the initial step of DRFI, the input image is classified into a couple of regions via a graph-based segmentation approach [14].

Then, based on the features like the histogram of RGB, HSV, and L*a*b values, Local Binary Patterns (LBP Features), and the response of Leung-Malik filters, regional contrast, backgroundness, and property descriptors are calculated for each k-th feature. Denoting v for the vector of features, according to each Region R_i, the first two descriptors are derived from the equations below, respectively:

$$x_k^c(R_i) = \sum_{j=1}^{M} \alpha_j \, \omega_{ij} \, D_k(v^{R_i}, v^{R_j}) \qquad (2)$$

$$x_k^b(R_i) = D_k(v^{R_i}, v^{R_j}) \qquad (3)$$

where $D_k(v^{R_i}, v^{R_j})$indicates the difference of the k-th feature of the regions R_i and R_j, ω_{ij} equals to $e^{-\frac{||p_i-p_j||^2}{2}}$, and α_j is defined as the normalized area of the region R_j. The property descriptors, which contains both appearance and geometric features, is extracted from [15].

These features and descriptors are used to train a random forest regressor whose purpose is to assign a saliency score to each discriminative region, based on their respective feature vector. The final trained model is reached to extract salient regions from the ones which are suspected to be background.

2.5. Seeded Region Growing

Seeded Region Growing was initially presented in 1994 for automotive segmentation of images in a number of regions based on some initial pixels, which will be replaced with initial clues when noise affects the input image [2]. In this method, all neighbors of the cues are investigated in several iterations, to have a label assigned to them. The process is repeated until all pixels of the input image are labelled.

As mentioned before, the main difference between the previously proposed SRG and the Deep Seeded Region Growing algorithm used in this work is the features to whom the expansion criteria are based on. In the previous work, the labelling criteria was established based on intensity rate, colour, standard deviation of the intensity pixels in the neighboring region of the spatial locations, and the location of edges and boundaries. In order to forge their segmentation network, Huang et al. [1] discarded these low-level features and used the high-level features derived from the network itself, instead. Normalized class activation maps represent these semantic features.

This algorithm expands the seeds based on verifying the neighbor pixels of the labelled cues. In each iteration, each pixel is received a label c, if satisfying these three conditions:

1. Being neighbor with at least one pixel having the same label c.
2. Having a value on the activation map of class c, passing a user-defined threshold Θ_f.
3. Having its maximum activation value among all objects and background classes, in the one belonging to class c.

In the previous SRG method [2], the labelling process was ran in the way that in finite number of iterations, all pixels were assigned to a cluster. However, in DSRG, seeding process might fail labelling all the pixels because of its strict criteria defined.

In the training process, while the SRG block sees its seed map unchanged in one iteration, it passed the expanded seed cues to the base network as annotation labels.

2.6. Loss function

After completing the process of training the network, a retraining process is started based on reaching activation maps and expanding the seeds as discussed before. In this step, the loss function aims to improve the segmentation accuracy, not the classification accuracy. In SEC [5], the final loss function is sum of three different functions, called “seeding loss”, “boundary loss”, and “expansion loss”. Seeding loss is the loss of activating seeds which is computed by comparing the outputs of CAM and SRG. Boundary loss is for being ensured that the boundaries of the objects are determined as well as the internal seeds. The latter function also prevents the seed regions to be grown uncontrollably.

In Huang et al. [1], only the first two functions are taken into account. These functions are calculated as follows:

i. Seeding loss

Seeding loss is balanced between object classes and background class. To some extents, the function is similar to categorical cross-entropy loss. Equation (4) shows this function, considering $H_{u,c}$ to be the activation map of the class *c* at the location *u*, and S_c to be the set of locations labelled as class *c*. (The set of all classes is defined as *C*.)

$$l_{seed} = -\frac{1}{\sum_{c\in C}|S_c|}\sum_{c\in C}\sum_{u\in S_c}\log\left(H_{u,c}\right) \qquad (4)$$

ii. Boundary loss

As of the previous method SEC [5], loss boundary is calculated throughout a Conditional Random Field (CRF) regression algorithm. The exact name of this type of loss in "Constrain-to-boundary" loss [16]. For computing this phrase of loss, initially the input image is downsampled to have a size equal to the output segmentation mask. Then, a fully-connected CRF, named as $Q(X, f(X))$ is constructed. The boundary loss is achieved from equation (5).

$$l_{boundary} = \frac{1}{n}\sum_{c\in C}\sum_{u\in 1} \quad Q_{u,c}\big(\mathrm{X}, \mathrm{f(X)}\big)\log\frac{Q_{u,c}\big(\mathrm{X,f(X)}\big)}{f_{u,c}(\mathrm{X})} \qquad (5)$$

CHAPTER 3: Experiments

3.1 Introduction

In the project, we trained a VGG-16 model pretrained on ImageNet data set [17], on a subset of the vehicle classification data set named “Miovision traffic camera dataset” (MIO-TCD) [18]. In addition, we evaluated CAM and DRFI on our data set. At the end of this chapter, we provide a brief discussion about pros and cons of these methods. Our final aim is to merge CAM and DRFI, and train the base network again, on MIO-TCD localization data set.

3.2 Data set

MIO-TCD is a data set published in 2017 and containing near 650 thousand classification images, and over 137 thousand localization images from each of vehicle types: “articulated truck, bicycle, bus, car, motorcycle, non-motorized vehicle, pickup truck, single unit truck, work van, and pedestrian” and a background class containing non-vehicle and non-pedestrian scenes of streets e.g. traffic lights, ground, tree leaves, and so on. In the initial steps, we aimed to use Cityscapes data set [19]. However, since our focus was on classifying and localizing vehicles, and after founding this data set, we changed our data set.

In order to reach acceptable initial results for a segmentation training, it was important to have a well-trained network on such data set. However, due to the time and memory limitations, we extracted a subset of the data set, including 1500 images of each class, except background class (because our purpose for background detection was based on DRFI algorithm). Figure 3.1 exemplifies on samples of MIO-TCD Data set, and in addition, Figure 3.2 reveals the accuracy results of classification on six famous networks, combined with linear SVM classifiers.

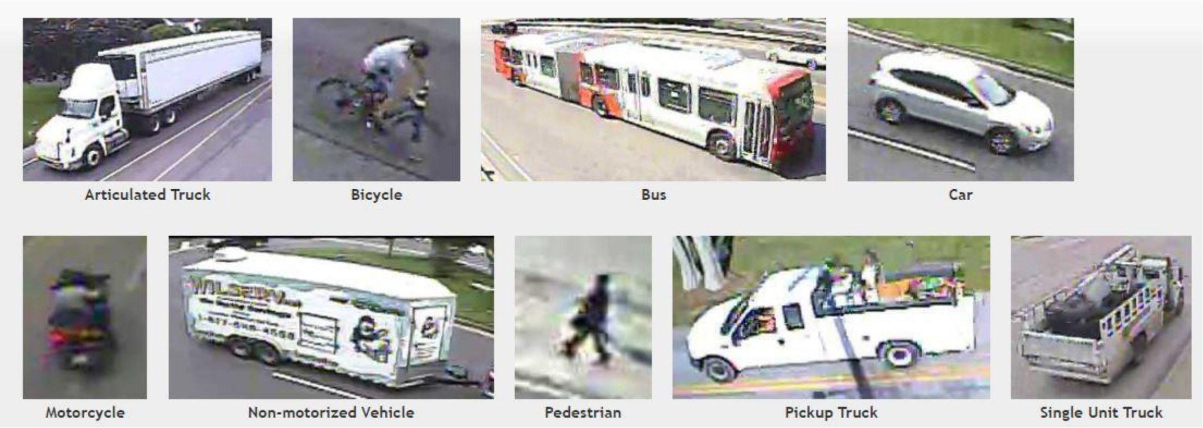
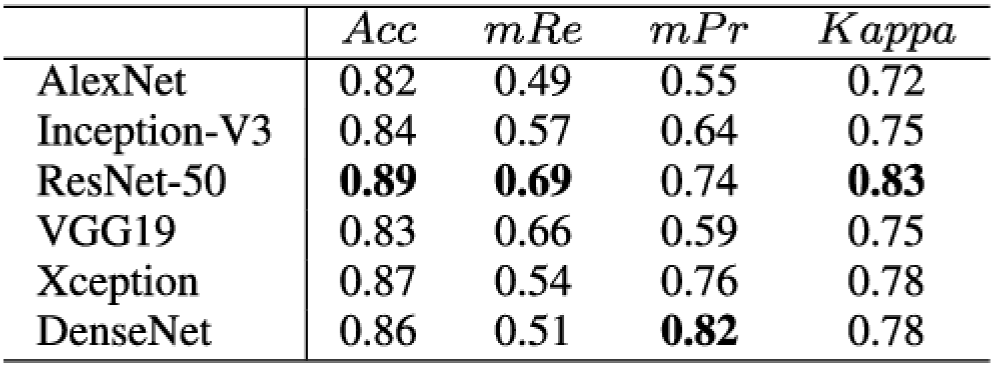


Figure 3.1: Instances of MIO-TCD data set.



	Acc	mRe	mPr	$Kappa$
AlexNet	0.82	0.49	0.55	0.72
Inception-V3	0.84	0.57	0.64	0.75
ResNet-50	**0.89**	**0.69**	0.74	**0.83**
VGG19	0.83	0.66	0.59	0.75
Xception	0.87	0.54	0.76	0.78
DenseNet	0.86	0.51	**0.82**	0.78

Figure 3.2: Classification Results of MIO-TCD on six networks [18].

$\boldsymbol{Acc} = Accuracy, \boldsymbol{mRe} = mean\ Recall, \boldsymbol{mPr} = mean\ Precision$

3.3 Model

Although no constrains are defined by the authors of DSRG [1] for the base network model, this should be incorporated that the networks in which the output layer is down sampled remarkably compared with the input images are expected to provide lower resolutions of CAMs, and consequently, lower segmentation accuracies. In the paper of DSRG, experiments are done with two networks VGG-16 and Resnet101. For progressing this project, we chose VGG-16 network, whose summarized structure is summarized in Figure 3.3. The reason behind this choice was the fact that for the same sizes of inputs, the last convolutional layer in VGG16 is twice larger in height and width, in comparison with Resnet101. This also should be noticed that VGG16 was pretrained on ImageNet.

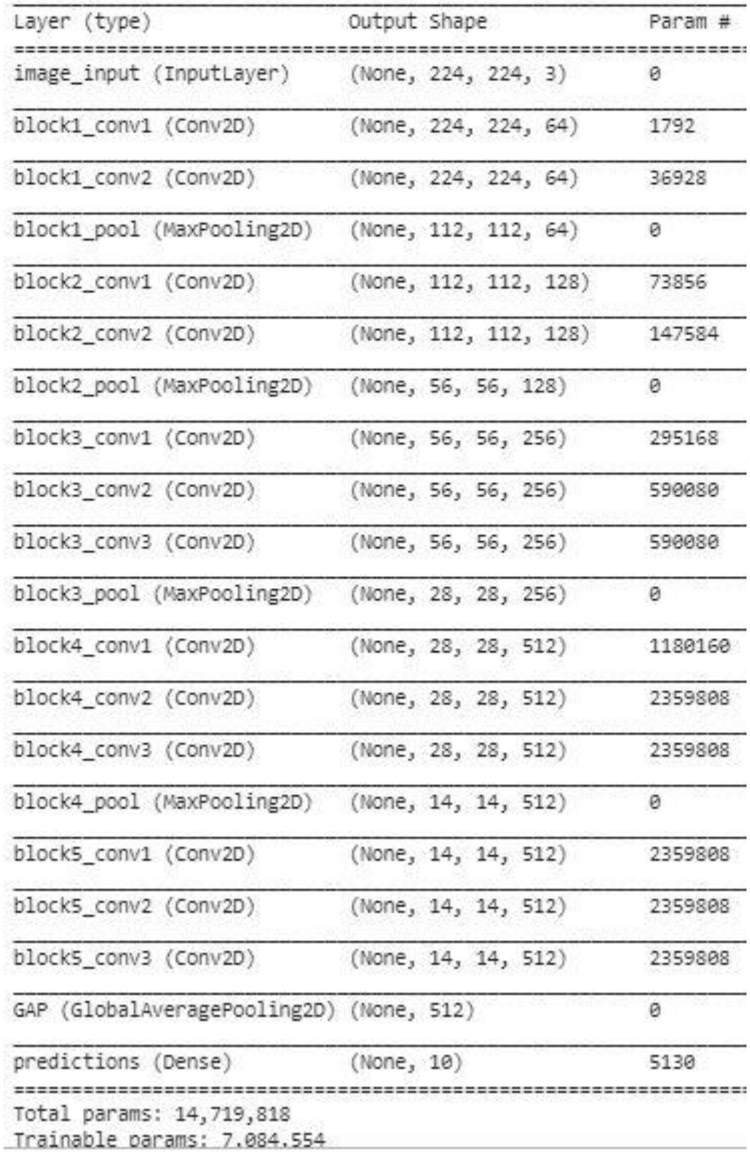
To be compatible for applying CAM, we did another alteration to the model as well: we replaced the classifier layers of VGG16 with a Global Average Pooling (GAP) layer, and an output layer, whose weighting values are initialized and need to be trained in the classification training phase, as discussed in section 2.3.

For the classification operation, our expectation was to reach a close accuracy to the one mentioned for VGG19, which is a deeper version of VGG16 (the difference between these two is the additional three convolutional layers that VGG19 has.) mentioned in Figure 3.2. However, as the training data in our work is much smaller than the one applied to VGG19 in benchmark [18], it was expected to obtain a test accuracy measure around 80%.

3.4 Training Algorithm

For doing the training process, the transfer learning technique was applied to the base network. Since the network pre-trained on a large data set like ImageNet which contains 1000 classes with multiple categories, the first 10 convolutional layers of this network are well-trained for detecting low and mid-level features. Our training is focused on making the network learning high-level features suitable for detecting vehicles, and classifying purposes. As can be seen in Figure 3.3, freezing all the convolutional layers except the latter 3 resulted to reduction of trainable parameters by more than half, which equals to a significant speed-up the training process.

Furthermore, for augmenting data, random processes like vertical and horizontal flipping, sheering, and rotating have been applied to the training data.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
image_input (InputLayer)     (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
GAP (GlobalAveragePooling2D) (None, 512)               0
_________________________________________________________________
predictions (Dense)          (None, 10)                5130
=================================================================
Total params: 14,719,818
Trainable params: 7,084,554
```

Figure 3.3: The summary of the architecture used in VGG16.

For the training process, we initiated the learning rate to 0.005, dividing this rate by half in every 10 epochs. Also, we deployed Adam optimizer, and splitted the used data subset in order to move 20 percent of objects in each image for testing, keeping the rest for being utilized in the training stage. The training has been finished at the end of epoch 30.

3.5 Training Results

After reaching a training accuracy of 92.19%, we validated our method on the test set. Figures 3.4 and 3.5, and Tables 3.1 and 3.2 summarize our classification results.

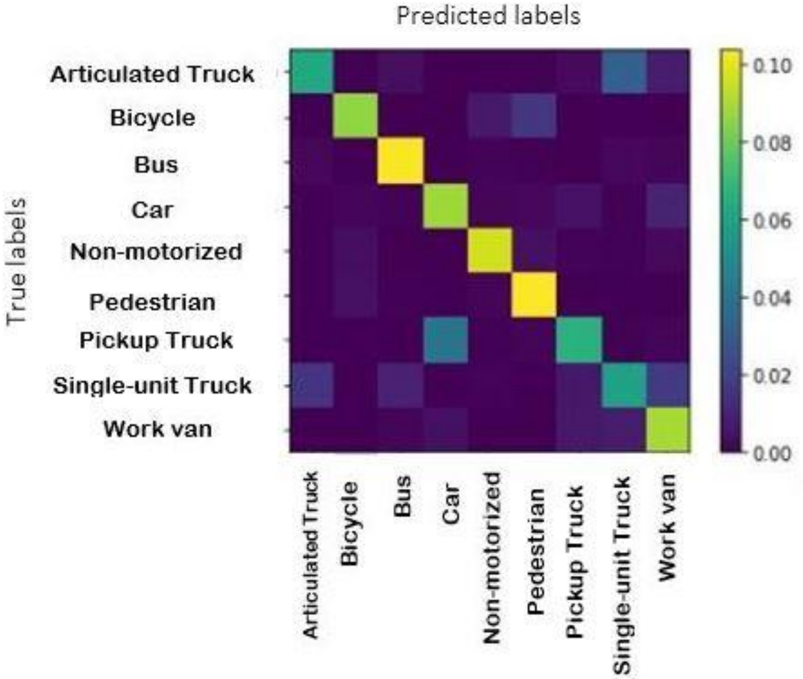


Figure 3.4: The Normalized Confusion Matrix of the classes.

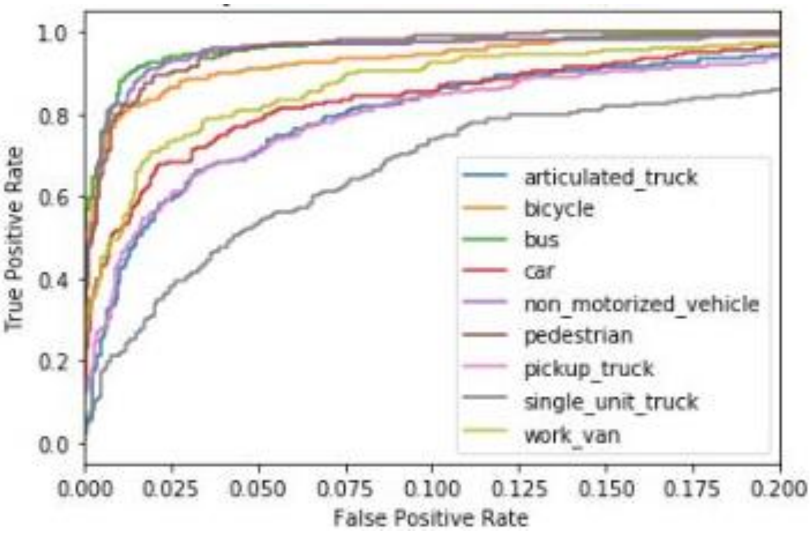


Figure 3.5: The joint ROC curve of the classes.

According the Figure 3.4 and Table 3.1, correct classification of single-unit trucks and articulated trucks are more challenging compared with the other classes because of the similarity of features and structures between these two classes, and also pickup truck, increasing misclassifications between these groups. Furthermore, the highest classification accuracy reach by pedestrian is possibly due to the major difference of the features of these instances compared with the other vehicle classes.

The F-1 scores computed and revealed in table 3.1 are achieved by the equations below:

$$P = \frac{TP}{TP+FP}, \quad R = \frac{TP}{TP+FN} \qquad (6)$$

$$F - 1\,score = \frac{2*P*R}{P+R} \qquad (7)$$

Where *P* and *R* represent the terms “precision” and “recall”, respectively, which are determined by the two basic terms, *TP* (True Positive), TN (True Negative), and *FN* (False Negative).

Class name	F-1 score	Class name	F-1 score
Articulated truck	0.5667	Pedestrian	0.93
Bicycle	0.7767	Pickup truck	0.6033
Bus	0.9233	Single-unit truck	0.5367
Car	0.8167	Work van	0.8067
Non-motorized vehicle	0.8784	***	***

Table 3.1: The F-1 scores of output classes

Test Accuracy	**78.56%**
Average Prediction time	267.04 ms

Table 3.2: The results of classification-training stage

3.6 Implementation of DRFI

Training of the saliency map detection method has been done using a subset of MSRA dataset [20], provided by Microsoft. After training with 3000 images of this data set which are belonged to natural and general scenarios, the trained model is employed in the proposed specific scenario, urban scenes, without any specific modification. This method helps the whole structure by discriminating the background regions from the possible candidates to be segmented as one of our goal classes. Therefore, well-performing this method can help the base network to run the segmentation process faster, and detect boundaries more precisely.

As mentioned in the benchmark provided in 2015 by A.Borji [21], DRFI was one of the most outperforming algorithms in salient object detection, by the time that the benchmark was published. However, it is concluded in the benchmark that there are some drawbacks with this method. The negative points are represented below:

- DRFI faces with difficulties while detecting smaller objects.
- This approach assumes that a dominating object is located in the image. (this negative point was not a case happening in our analysis, as we derived our results by giving input images from MIOTCD [18].
- The average runtime of DRFI was reported as 0.697 seconds in [21], while in our experiments, we achieved an average runtime of 1.24 seconds, making the algorithm a little slow for real-time applications.

(that might be a case of difference of devices, as we utilized an Intel i7 2.40GHz CPU with 4GB RAM.) We visualized a sample of our performance results in Figure 3.6 and 3.7, which are our success and failure cases, respectively. The failure cases for this saliency maps were primarily belonged to pedestrians, as of their small sizes putting the saliency model into challenge. That is while the higher classification results were belonged to the pedestrian class in our classification stage.

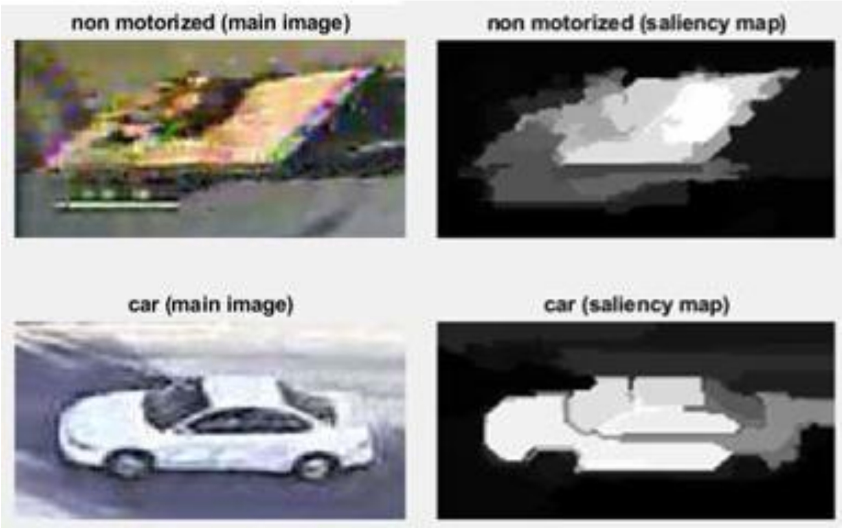


Figure 3.6: The success cases of DRFI method.

Unfortunately, as we chose a data set which does not contain any localization annotations, we could not add a mIOU (mean Intersection over Union) metric to our report.

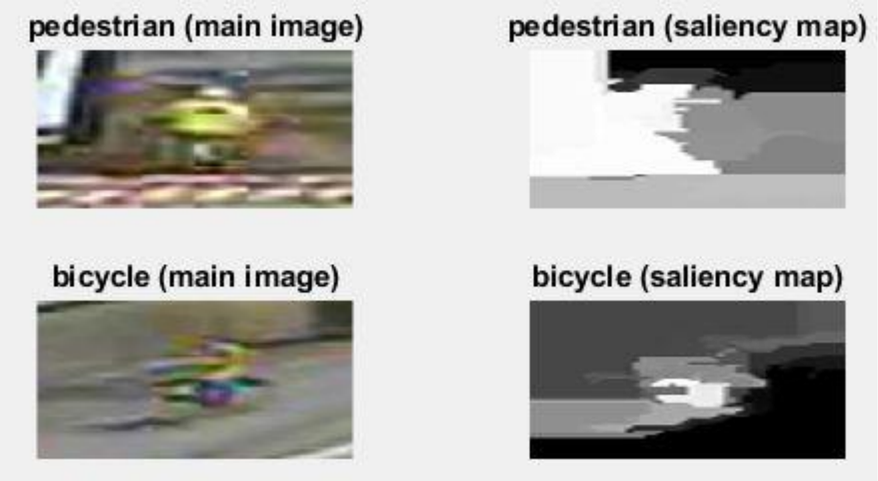


Figure 3.7: The failure cases of DRFI method

3.7 Initialized seed cues

As discussed before, initial seeds are derived from the core network itself, by multiplying the feature maps of the last convolutional layer with the weighting factors behind the output layer. Here are some sample results of CAM method provided in Figure 3.8.

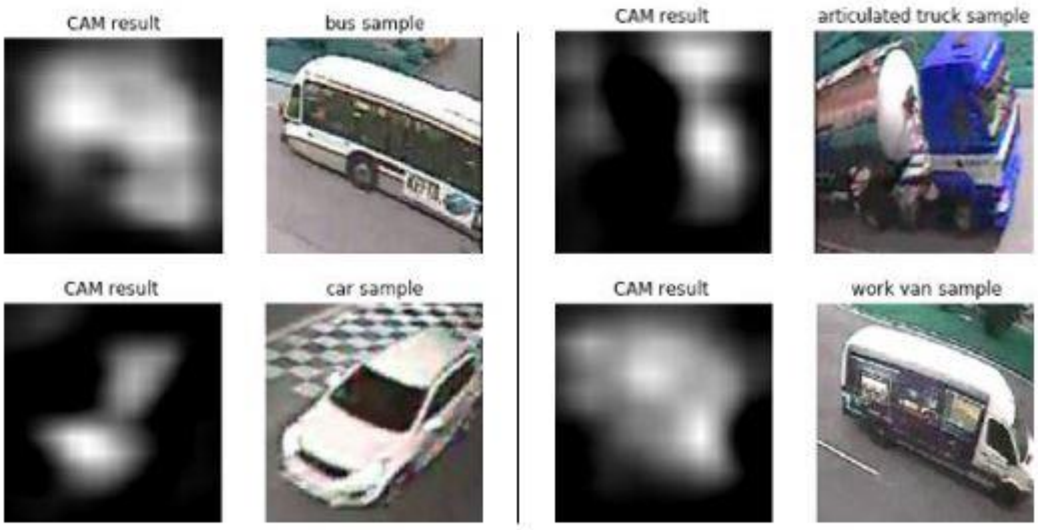


Figure 3.8: The Class activation maps derived from the data

Referencing to the main method [1], by applying a hard threshold on the outputs of CAMs, seeding cues are initialized for object classes. The same thing happens for background cues, by applying a separate hard threshold on the negative of the saliency map achieved before.

CHAPTER 4: Conclusion & Future work

4.1 General Conclusion

In this work, we made our focus on applying DSRG semantic segmentation approach on localizing vehicles, which is a functional task in self-driving cars, traffic cameras, and etc. Being aware that each single false prediction in these applications can cause catastrophic consequences like causing fatal accidents and letting the careless drivers bypass the law, we aimed to provide pros and cons with the blocks, while implementing and getting results from them. Instead of using a trained model for our main purpose, we aimed to retrain the network with a classification result to have the segmentation results improved. We showed that the last three convolutional layers of our chosen network, VGG-16 were adopted to extracting the high-level features related to different instances of vehicles.

In the classification stage, our results could get more precise if we would use more data sets for each class, also increasing the images in car classes in comparison with the other classes as far as cars are more common in streets and roads rather than the other kinds of vehicles. Although the relatively low resolution of the images in MIO-TCD had impact in making classification more challenging, we also could have verified whether overfitting has occurred in our network to some extents or not. Such analysis might have been resulted to use a network trained with less number of epochs.

About the implementation of DRFI and CAM, pros and cons can be summarized for both processing blocks in this paragraph as follows: in performance, both methods are able to extract accurate cues for large object and background regions. In addition, both of these algorithms ease the process of segmentation by representing high-level semantic features. However, the runtime of DRFI is a little slow. This disadvantage might cause challenges for the network in the real-time applications like autonomous driving. Moreover, the failure rate increase in both of these methods while dealing with smaller objects or areas. In our experiment, the resolution of Activation Maps for class instances is divided by 16 with respect to the input size. In the segmentation stage, it is predicted for classes “pedestrian” and maybe “bicycle” to have the most failure segmentation rates (e.g. less mIOU metrics).

4.2 Future Work

In this work, we went throughout training the base network, and reaching the results from CAM and DRFI. In the next step, we will combine these two algorithms and make the second stage of our planned training, which is based on segmentation. For this stage, we will use MIO-TCD localization data set, which contains scenes with one or more vehicle instances. Furthermore, we might use the dataset Cityscapes [19] for improving our segmentation quality. In this stage, the role of SRG block can be highlighted. In this upcoming step, we expect the classification accuracy to drop insignificantly. However, the representations of the core network will be enhanced in the way that class activation maps shown in figure 3.8 are trained to point on the pixels which are really related to the classes whose corresponding neuron in the network has been activated. For future progresses, trying some alterations in the structure of algorithm is also incorporated, such as replacing CAM with Grad-CAM [22] explainers, which operates by calculating the gradient of output activation values with respect to the feature maps of the last convolutional layer.

REFERENCES

[1] Huang, Zilong, et al. "Weakly-supervised semantic segmentation network with deep seeded region growing." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[2] R. Adams and L. Bischof. Seeded region growing. IEEE TPAMI, 16(6):641–647, 1994.

[3] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. International Conference on Learning Representations, 2015.

[4] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In Proc. CVPR, pages 2921–2929, 2016.

[5] A. Kolesnikov and C. H. Lampert. Seed, expand and constrain: Three principles for weakly-supervised image segmentation. In Proc. ECCV, pages 695–711. Springer, 2016.

[6] Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost: Joint appearance, shape and context modelling for multi-class object recognition and segmentation. In: ECCV

[7] Carreira, J., Sminchisescu, C.: CPMC: Automatic object segmentation using con-strained parametric min-cuts. IEEE T-PAMI 34(7) (2012)

[8] Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., Belongie, S.: Objects in context. In: ICCV (2007)

[9] Nowozin, S., Gehler, P.V., Lampert, C.H.: On parameter learning in CRF-based approaches to object class image segmentation. In: ECCV (2010)

[10] He, X., Zemel, R.S.: Learning hybrid models for image annotation with partially labeled data. In: NIPS (2009)

[11] Dai, J., He, K., Sun, J.: BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: ICCV (2015)

[12] Pourian, N., Karthikeyan, S., Manjunath, B.: Weakly supervised graph based se-mantic segmentation by learning communities of image-parts. In: CVPR (2015)

[13] Vezhnevets, A., Ferrari, V., Buhmann, J.M.: Weakly supervised structured output learning for semantic segmentation. In: CVPR (2012)

[14] H. Jiang, J. Wang, Z. Yuan, Y. Wu, N. Zheng, and S. Li. Salient object detection: A discriminative regional feature integration approach. In Proc. CVPR, pages 2083–2090, 2013.

[15] D. Hoiem, A. A. Efros, and M. Hebert, “Geometric contextfrom a single image,” inICCV, 2005, pp. 654–661

[16] Krahenbuhl, P., Koltun, V.: Efficient inference in fully connected CRFs with gaussian edge potentials. In: NIPS (2011)

[17] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[18] Luo, Zhiming, et al. "MIO-TCD: A new benchmark dataset for vehicle classification and localization." IEEE Transactions on Image Processing 27.10 (2018): 5129-5141.

[19] Cordts, Marius, et al. "The cityscapes dataset for semantic urban scene understanding." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[20] Borji, Ali. "What is a salient object? A dataset and a baseline model for salient object detection." IEEE Transactions on Image Processing 24.2 (2014): 742-756.

[21] Borji, Ali, et al. "Salient object detection: A benchmark." IEEE transactions on image processing 24.12 (2015): 5706-5722.

[22] Selvaraju, Ramprasaath R., et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization." Proceedings of the IEEE International Conference on Computer Vision. 2017.