Multimodal Convolutional Neural Network for Object Detection Using RGB-D Images

Irina Mocanu
Computer Science Department
University Politehnica of Bucharest
Bucharest, Romania
irina.mocanu@cs.pub.ro

Cosmin Clapon
Computer Science Department
University Politehnica of Bucharest
Bucharest, Romania
cosmin.clapon@cs.pub.ro

***Abstract*—This paper presents a new convolutional neural network architecture for performing object detection based on RGB-D images. The network is an extension of the Faster-RCNN network where we add an additional input network branch for processing the depth image. The network was evaluated on the SUN RGB-D dataset for object detection and we obtained a positive difference in mAP score of about 4%, compared to the original one.**

***Index Terms*—object detection, convolutional neural network, RGB-D images**

I. Introduction

Visual recognition is an important problem in computer vision with applications ranging from robotics, autonomous driving, object tracking, video surveillance and augmented reality. Many recent state-of-the-art systems for visual recognition focus on RGB images only, but new sensing technology, such as Kinect sensors, that can record high quality RGB-D data (RGB and depth images) has become affordable and ubiquitous in many robotic systems.

The development of robotic software that allows reasoning and mimic of human-like behavior opens a world of possibilities where robots are potentially used in multiple ecosystems for human aid and assistance. The main challenges occurring are the safety assurance and the avoidance of the risk of accidents. One of the main problems in this regard is the safety of object manipulation in robot-assisted environments for humans. Thus, the main problem consists in recognizing different objects and learning a way to grab and hold them.

This paper proposes an object recognition convolutional neural network performing object recognition from RGB-D images. Compared to RGB images, which provide useful information about texture and appearance, the depth image is invariant to lighting or color variations. Also it provides useful additional information about object geometry and allows better segmentation of objects from background.

The rest of the paper is organised as follows: Section II describes related work. Section III gives the description of the proposed architecture. Evaluation of the proposed network are explained in Section IV. Conclusions and future work are given in Section V.

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CCCDI UEFISCDI and of the AAL Programme with co-funding from the European Unions Horizon 2020 research and innovation programme project "IONIS - Improving the quality of life of people with dementia and disabled persons", project number AAL-2016–074—IONIS-2 within PNCDI III.

II. Related Work

There are two main methods for performing object recognition: (1) conventional methods based on hand crafted features that capture visual appearance and (2) methods based on convolutional neural networks. From the first class methods, paper [1] combines RGB and depth features. They use spin images [2] and SIFT descriptors. Spin images are computed on randomly sub-sampled subsets of 3D points where each image is centered on a 3D points and captures the spatial distribution of points within its own neighborhood. Based on spin images and SIFT features, efficient match kernel (EMK) features [3] are computed. After that principal component analysis is applied in order to select a number of components.

In the last few years, convolutional neural networks (CNNs) have been used in the context of visual recognition. CNNs have the advantage of not requiring hand crafted feature representations for the image. They derive the feature space representations directly from the data by training them on large datasets. Important progress were made in image classification and recognition using CNNs architectures like AlexNet [4], VGG [5], GoogLeNet [6], ResNet [7] which perform very well on large RGB image datasets. For object detection, several network architectures have been proposed. For example, method described in paper [8] outperforms methods based on traditional hand-crafted features. Most of these models were trained on RGB datasets such as PASCAL VOC to predict objects and corresponding bounding boxes of a proposed image region. Also, CNNs were also proposed on RGB-D images in order to perform bounding box detection [9]. In [10] it is proposed an approach in which the depth is encoded with three channels where each pixel encodes the horizontal disparity, height above ground and the angle between the pixels surface normal vector and the gravity vector. Finally, they scale linearly the values across the whole dataset to 0-255 range. Another approach - multi-modal deep feature, described in [11], allows learning modality-correlated and modality-specific feature representations. They used shared

weights strategy and a parameter-free correlation layer for learning modality-correlated features.

This paper presents a new CNNs architecture - Multimodal Faster-RCNN- by combining RGB and depth images. The architecture extends the Faster-RCNN network proposed in [12] with a depth branch - outperforming its performances tested on SUN RGB-D dataset [13].

III. Description

Faster-RCNN architecture [12], a recent result in CNNs for object detection, based on RGB images is extended with an additional input network branch which processes the depth image. The network performs object detection taking as input RGB and depth images (Figure 1). There are cases where color information is not enough for good object detections. For example, there are objects that confound with the background of non-object regions in the color domain. These cases can be much more visible in the depth image given that they have enough geometric structure. Also, the object boundaries are much easier to be identified into depth images and this can lead to more accurate boundaries for the object detections. We assume they are of the same size and they are properly aligned such that any pixel in the RGB image corresponds to a pixel in the depth image at the same location. A region in the RGB-D image is defined by a 2D bounding box parameterized by a 4 dimensional vector (x; y; w; h), where (x, y), w and h denote the box's center coordinates and its width and height.

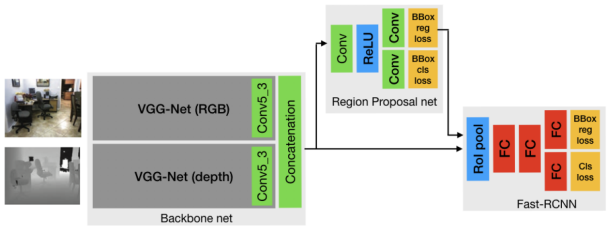

Fig. 1. Proposed CNN.

Faster-RCNN is composed of two parts: (1) Region Proposal Network (RPN) which is responsible for producing object region proposals and (2) Fast-RCNN [14] that uses the proposed regions to further refine them in order to perform object detection. The RPN is a fully convolutional network that simultaneously predicts object bounding and its associated score. The proposed regions are used by the Fast-RCNN for detection. The key aspect of Fast-RCNN is that both networks take as input the same feature maps produced by a shared convolutional backbone network (VGG-Net [15]), thus obtaining a unified network that can be implemented efficiently and trained end-to-end. We modify the Faster-RCNN architecture by adding an additional input - the depth image and we named it multimodal Faster-RCNN. We add an additional input branch in the backbone network which computes feature maps for a depth image. The feature maps of the RGB and depth images, respectively, are concatenated and propagated through the original RPN and Fast-RCNN. The depth branch is based on the VGG-Net architecture, but is fine-tuned with depth images.

We need to encode the depth image into a 3-channel image in order to be able to use it as input in VGG-Net pre-trained on ImageNet. For encoding the depth image into a 3-channel image, we use the method proposed in [16] which was used in a similar network architecture trained for object recognition in RGB-D images, and they found it was the most effective method compared to other commonly used depth encoding methods. Thus, the values of the depth map are normalized in the interval [0; 255]. After this step, a jet colormap is applied on the input image, which transforms it from a single to a three channel image by essentially colorizing the depth image. For each pixel in the depth map, the distance values are mapped to color values ranging from red to blue, distributing the depth information over all three RGB channels.

The RPN used in Faster-RCNN takes as input the convolutional feature maps of the backbone network and outputs a set of object proposals, each with its associated score. The RPN is modeled as a fully convolutional network, by implementing a sliding window approach over the convolutional feature map output by the backbone network. Each sliding window is mapped to a 5*512-d feature vector and fed into two fully connected layers, a bounding box regression layer and a box classification layer. The sliding window is 3x3 and the fully-connected layers are shared accross all spatial locations. This approach is naturally implemented with a 3x3 convolutional layer followed by two 1x1 convolutional layers (one for box regression and another for box classification) (see Table I).

TABLE I
RPN Layers

Type	Size	No outputs	Stride	Padding
Conv	3x3	2*512	[1,1]	[1,1]
ReLU	-	-	-	-
Conv-reg	1x1	4k	[1,1]	[1,1]
Conv-cls	1x1	2k	[1,1]	[1,1]

The outputs of the two layers are 3D tensors which contain k-dimensional vectors for each sliding window location. Each vector contains k object proposals. The box regression layer has 4k outputs encoding the coordinates of k boxes, and the classification layer outputs 2k scores that estimate a probability distribution which tells if the proposal is an object or a non-object. The proposals are relative to k reference boxes, which are called anchors.

The loss for training the RPN network is defined as

$$L_{RPN} = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i; p\ ) + \frac{1}{N_{reg}} \sum_i p_i\ L_{reg}(t_i; t_i) \quad (1)$$

where L_{cls} is the classification loss defined as the log loss over two classes (object or non-object), and L_{reg} is the bounding box regression loss defined as the smooth $L1$ loss, i is the index of an anchor and p_i is the predicted probability of anchor i being an object. p_i is the ground-truth label and is set to 1 if the anchor is positive, and 0 if the anchor is negative.

A positive anchor is the anchor with highest Intersection-over-Union (IoU) overlap with a ground-truth box, or an achor that has an IoU overlap higher than 0.7 with any ground-truth box. Negative anchors are anchors with IoU ratio lower than 0.3 for all ground-truth boxes. Anchors that do not meet any of the above conditions (positive or negative) are not counted in the loss objective. t_i is the ground-truth box represented by a 4-dimensional vector with the box's coordinates (center coordinate, width, height). N_{cls}, N_{reg} and are normalization parameters. We set = 10 and the other two parameters depend on the training batch size and the number of anchor locations, respectively.

Fast-RCNN is responsible for detecting the objects in region proposed by the RPN network. It takes as input the proposals and outputs for each proposal a class label and a bounding box. The proposals produced by the RPN network are relative to the convolutional feature maps produced by the backbone network. These feature maps are already computed for the RPN network, so we can reuse them without additional computation. Because we don't recompute the convolutional feature maps for each image region, we need a method to extract from the convolutional feature maps the regions that are associated to each box proposal.

The Region of Interest (RoI) pooling layer is special type of layer used introduced in the Fast R-CNN paper. Its a layer that uses the max pooling operation to convert features inside the region proposal into a small feature map with a fixed size of HxW, which are independent of the box size. The RoI pooling layer acts upon the convolutional feature maps produced by the backbone network. First the RoI's coordinates are projected from the input image space to the convolutional feature maps space. Then each RoI is divided into an HxW grid of sub-windows and max pooling is applied in each sub-window. The pooling operation is applied to each feature map channel independently. We set *H=7* and *W=7*. The RoI pooling operation is performed for each proposal output by the RPN. Then, the feature maps are fed into a sequence of two fully connected layers which produce feature vectors for a given proposal. This feature vector is further used in two output fully connected layers which perform bounding box regression and classification (Table II).

TABLE II
FAST-RCNN LAYERS

Type	Size	No outputs	Stride	Padding
FC	4096	4096	-	-
FC	4096	4096	-	-
FC-cls	K	K	-	-
FC-reg	4K	4K	-	-

The loss for training Fast-RCNN is defined as:

$$L_{FAST\ \ RCNN} = L_{cls}(p_i; p\ ) + \qquad L_{reg}(t_i; t_i) \qquad (2)$$

where L_{cls} is the classification loss defined as the log loss. p is a discrete probability distribution over *K+1* classes (*K* object classes + background class); p is the ground truth one-hot vector representing the class probability distribution. For bounding boss regression loss L_{reg} we use the smooth *L1* loss and is evaluated only if the proposal is not a background proposal, otherwise it's set to 0; t and t are vector parameterizations of the predicted bounding boxes and the ground truths, respectively. They are defined the same way as in the RPN network and it is set to 1.

IV. MULTIMODAL FAST-RCNN EVALUATION

We trained and evaluated the Multi-Modal Fast-RCNN network for 2D object detection on the SUN RGB-D dataset [13] that is a RGB-D dataset containing 10.335 RGB-D images of indoor scenes with annotations in both 2D and 3D for the objects present in the scene. The authors of SUN RGB-D capture a significant amount of new data by themselves and also combine existing RGB-D datasets. They capture using Kinect v2 3748 images, and 1159 images using Intel RealSense. They included 1449 images from the NYU Depth V2 [5] dataset, and added 554 realistic images from the Berkeley B3DO [17] dataset. Both of these datasets are captured with a Kinect v1. They also manually selected 3389 frames from SUN3D [18] videos captured by Asus Xtion. The total amount of images for the entire SUN RGB-D dataset is 10335 RGB-D frames.

For training the multimodal Fast-RCNN network, we use an alternating training algorithm consisting of three steps. In the first two steps, the goal is to fine-tune each input branch for each modality (RGB and depth). More specifically, in the first step we initialize the RGB input branch with VGG-Net weights pre-trained on ImageNet and train the whole network. We fix only the first two sets of convolutional layers of the VGG-net network (RGB branch) and let the rest of the network to be trained. The whole VGG-Net of the depth branch is fixed. In the second step, we fine-tune the depth branch in a similar manner, keeping the RGB input branch fixed and initialized with the weights computed at the first step. Finally, in the third step we fix both input branches and fine-tune the other two networks (RPN and Fast-RCNN). For the RPN network, we use the original parameters for anchors, i.e. 3 scales with box areas 1282, 2562, 5122 pixels, and 3 aspect ratios of 1:1, 1:2, 2:1. The input images are scaled such that their width is 600 pixels. We use a single image per iteration and perform for each one of the training steps 40,000 iterations. For the RPN network, we randomly sample 256 positive and negative anchors roughly equal in proportion. If the number of positive anchors is not enough, we pad the batch with negative anchors. We use a learning rate of 0.001.

We rescaled the original images (both RGB and depth) from 640 x 480 to 1000 x 600. We have used 3 scales for anchors with box areas of 128*128, 256*256 and 512*512, with 3 aspect ratios of 1:1, 1:2 and 2:1, thus having 9 anchors for each position in the image with a stride (step) of 16. Given that the last convolutional layer of VGG outputs 60 x 40 feature maps and we use 9 anchors per position, there will be roughly 20000 (60 x 40 x 9) anchors (proposals). The anchor boxes that cross the image margins are ignored when computing the loss of the network, so there are about 6000 anchors per image during training the region proposal network.