An Approach for Multi Label Image Classification Using Single Label Convolutional Neural Network Classifier and Image Segmentation with Objectness Measure and Selective Search

Dr. Kazi Md. Rokibul Alam*, Setu Basak†, Shubhashis Karmakar‡
Department of Computer Science and Engineering, Khulna University of Engineering & Technology
Khulna, Bangladesh
Email: *rokibcse@yahoo.com, †setu.1421@yahoo.com, ‡shubhashis@gmail.com

***Abstract*—Single label image classification has been promisingly demonstrated using Convolutional Neural Network (CNN). However, how this CNN will fit for multi-label images is still difficult to solve. It is mainly difficult due to the complex underlying object layouts and insufficient multi-label training images. This work has proposed an approach for classifying multi-label image by a trained single label classifier using CNN with objectness measure and selective search. We have taken two established image segmentation techniques for segmenting a single image which is a multi-label image into some segmented images. Then we have forwarded the images to our trained CNN and predicted the labels of the segmented images by generalizing the result. Our single-label image classifier gives 87% accuracy on CIFAR-10 dataset. Using objectness measure with CNN gives us 51% accuracy on a multi-label dataset and gives upto 57% accuracy using selective search both considering top-4 labels which is significantly good for a simple approach rather than a complex approach of multi-label classifier using CNN.**

Keywords*—*Image Classification, CNN, Objectness Measure, Selective Search.

I. Introduction

In the computer vision field, there has been increasing interest in object category recognition. Among the major tasks, image classification is defined as the task of assigning an image one or multiple labels corresponding to the presence of a category in the image.

Multi-label image classification is however a more general and practical problem, since the majority of real-world images are with more than one objects of different categories. Many methods have been proposed to address this more challenging problem. The success of CNN on single-label image classification also sheds some light on the multi-label image classification problem. However, the CNN model cannot be trivially extended to cope with the multi-label image classification problem in an interpretable manner, mainly due to the following reasons. Firstly, the implicit assumption that foreground objects are roughly aligned, which is usually true for single-label images, does not always hold for multi-label images. Such alignment facilitates the design of the convolution and pooling infrastructure of CNN for single-label image classification. However, for a typical multi-label image, different categories of objects are located at various positions with different scales and poses. For example, as shown in Figure 1, for single-label images, the foreground objects are roughly aligned, while for multi-label images, even with the same label, i.e., cat and bird, the spatial arrangements of the cat and bird instances vary largely among different images. Secondly, the interaction between different objects in multi-label images, like partial visibility and occlusion, also poses a great challenge. Therefore, directly applying the original CNN structure for multi-label image classification is not feasible. Thirdly, due to the tremendous parameters to be learned for CNN, a large number of training images are required for the model training. Furthermore, from single-label to multi-label (with n category labels) image classification, the label space has been expanded from n to 2n, thus more training data is required to cover the whole label space. For single-label images, it is practically easy to collect and annotate the images. However, the burden of collection and annotation for a large scale multi-label image dataset is generally extremely high.

(a) a (b) b (c) c (d) d

Fig. 1: (a), (b) Some examples from CIFAR-10 [4]. The objects in single-label images are usually roughly aligned.(c),(d) However, the assumption of object alignment is not valid for multi-label images. Also note the partial visibility and occlusion between objects in the multi-label images.

So we propose a simple technique using our trained single-label classifier of CNN with the objectness measure [1] and selective search [2]. First we segment a multi-label image into some segments or image windows using the two approaches and then test these images against our trained single label model and predict the multiple labels of the image. We have taken different approaches like taking top-1, top-2, total sum of the scores and cumulative percentage sum of the scores to predict the labels.

II. Related Works

During the past few years, many works on various single-label and multi-label image classification models have been

conducted. These models are generally based on deep learning.

A. Deep Learning Based Models

Deep learning tries to model the high-level abstractions of visual data by using architectures composed of multiple non-linear transformations. Specifically, deep convolutional neural network (CNN) has demonstrated an extraordinary ability for image classification on single-label datasets such as CIFAR-10/100 [4] and ImageNet [3]. A. Krizhevsky et al. [3] trained one of the largest convolutional neural networks to date on the subsets of ImageNet used in the ILSVRC-2010 and ILSVRC-2012 competitions. Their network contains a number of new and unusual features which improve its performance. The size of their network made overfitting a significant problem, even with 1.2 million labeled training examples, so they used several effective techniques for preventing overfitting. Their final network contains five convolutional and three fully-connected layers, and this depth seems to be important: they found that removing any convolutional layer (each of which contains no more than 1% of the models parameters) resulted in inferior performance.
But their network is huge and complex. Their networks size is limited mainly by the amount of memory available on current GPUs and by the amount of training time that they are willing to tolerate. Their network takes between five and six days to train on two GTX 580 3GB GPUs. All of their experiments suggest that their results can be improved simply by waiting for faster GPUs and bigger datasets to become available.
More recently, CNN architectures have been adopted to address multi-label problems. Gong et al. [5] studied and compared several multi-label loss functions for the multi-label annotation problem based on a similar network structure to [3].
However, due to the large number of parameters to be learned for CNN, an effective model requires lots of training samples. Therefore, training a task-specific convolutional neural network is not applicable on datasets with limited numbers of training samples.

III. Dataset

A. CIFAR-10 Dataset

The CIFAR-10 dataset is a subset of the 80 million tiny images. They were collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The CIFAR-10 dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class. The sample from CIFAR-10 dataset is given in Figure 2.

B. Multi Label Dataset

As we are using single label dataset(CIFAR-10) for our single label Convolutional Neural Network, we have to use images that contain the identical contents from CIFAR-10.

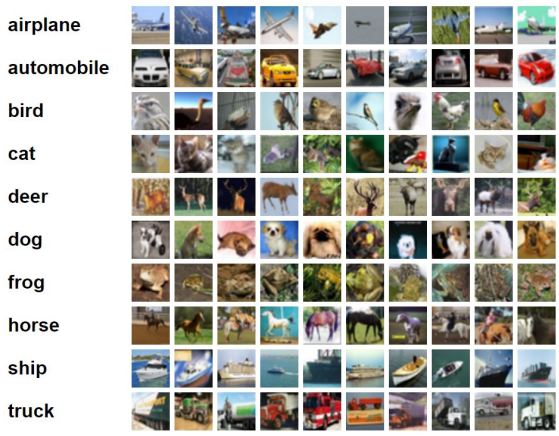


Fig. 2: CIFAR-10 Dataset

So, we picked random 100 images containing the identical contents as CIFAR-10 and tested it with our model. For an example, from Figure 3 we see that the image contains only a cat and a dog, which our Single Label Convolutional Network can classify. So, the images we chose only contains subset of airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck.

Fig. 3: Multi Label Image

IV. Architecture

A. Architecture for classifying single label images

Figure 4 shows the Convolutional Neural Network that classifies single label images.In the each layer of Convolutional Neural Network from the Figure 4, the input image is converted by Spatial Convolution(number of input channels $\rightarrow$ number of output channels, kernel height x kernel width, step height, step width, padding height, padding width). Then Spatial Batch Normalization is applied. After that Spatial Max Pooling(filter height, filter width, step height, step width) is used to downsample the images. Dropout is used to reduce overfitting. Rectifier Linear Unit is used for activation function. Also Linear Transformation(number of input channels $\rightarrow$ number of output channels) is used to reduce the number of channels by applying linear transformation.

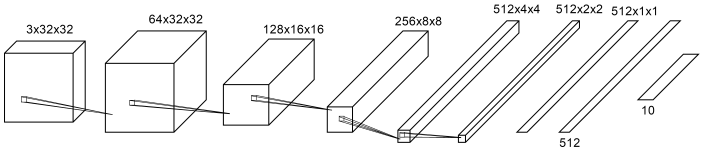


Fig. 4: Convolutional Neural Network for Classifying Single Label Images

a) Spatial Convolution: Applies a 2D convolution over an input image composed of several input planes. For an example, Spatial Convolution(3 $\rightarrow$ 64, 3x3, 1,1, 1,1) means that number of input channel is 3, and the number of output channel is 64. The kernel size is 3x3 and as there are 64 output channels, there will be 64 kernels, each having dimension 3x3. The step of the convolution is 1 step for height and width. The padding is 1 for height and width. Let's visualize this. In the figure 5 the input neurons are the ones representing each pixel of an image, the kernel size or the filter size is 5x5. This filter would iterate over every single pixel and for each pixel look at the 5x5 neighborhood and produce corresponding images. As the number of output channel is 64, this will be done 64 times.

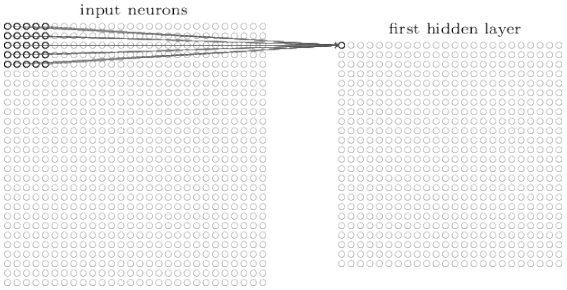


Fig. 5: Spatial Convolution with Kernel Size 5x5

b) Spatial Batch Normalization: Implements Batch Normalization as described in the paper [6]. The operation implemented is:

$$y = \frac{x - mean(x)}{standard\text{–}deviation(x)} * gamma + beta$$

where the mean and standard-deviation are calculated per feature-map over the mini-batches and pixels and where gamma and beta are learnable parameter vectors of size N (where N = number of feature maps).

c) ReLU: It is the activation function defined as:

$$f(x) = max(0, x)$$

.

d) Spatial Max Pooling: Applies 2D max-pooling operation. For example, Spatial Max Pooling(2,2,2,2) means that the max pooling will be done with filter 2x2 with step 2 in height and step 2 in width direction. Max pooling means that we will select the maximum value from 2x2 filter or the input area.

e) Dropout: Dropout is used to prevent the neural network from overfitting. Dropout is implemented by only keeping a neuron active with some probability p (a hyper-parameter), or setting it to zero otherwise. The input neuron is scaled by $1/p$ if it is not deactivated.

f) Linear: Applies a linear transformation to the incoming data ($y = mx + c$). For example, Linear(512 $\rightarrow$ 10) means that there are 512 input channels and these 512 channels are converted to 10 channels. So, the weight matrix will be 10x512.

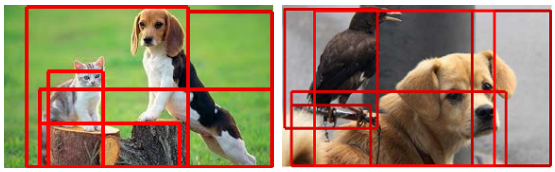
g) Loss Function: Cross-entropy loss function has been used which has the form:

$$L_i = -\log\left(\frac{e^{f_y}}{\sum_j e^{f_j}}\right)$$

h) Normalization on CIFAR-10: Normalization is required so that all the inputs are at a comparable range. This can be done to force the input values to a certain range. The images were converted from RGB channel to YUV channel. Then U and V channels were normalized globally with mean and standard deviation. The Y channel was normalized locally.

B. Image Segmentation

1) Measuring the objectness of image windows: B. Alexe et al. [1] presented a generic objectness measure, quantifying how likely it is for an image window to contain an object of any class. They explicitly trained it to distinguish objects with a well-defined boundary in space, such as cows and telephones, from amorphous background elements, such as grass and road. The measure combines in a Bayesian framework several image cues measuring characteristics of objects, such as appearing from their surroundings and having a closed boundary. These include an innovative cue to measure the closed boundary characteristic. Finally, they presented two applications of objectness. In the first, they sample a small number windows according to their objectness probability and gave an algorithm to employ them as location priors for modern class-specific object detectors. As they showed experimentally, this greatly



(a) a (b) b

Fig. 6: Segmented Images

reduces the number of windows evaluated by the expensive class-specific model. In the second application, they used objectness as a complementary score in addition to the class-specific model, which leads to fewer false positives. As shown in several recent papers, objectness can act as a valuable focus of attention mechanism in many other applications operating on image windows, including weakly supervised learning of object categories, unsupervised pixelwise segmentation, and object tracking in video. Computing objectness is very efficient and takes only about 4 sec. per image. This technique finds out some image windows like Figure 6.
The general scheme for using their objectness measure as a location prior for object detectors is algorithm 1. The algorithm inputs the class-specific confidence function c which the detector employs to score a window. They build an initial set l of $F = 1000$ windows multinomially sampled from the distribution D of windows scored by their objectness measure (Multi-scale Saliency)MS +(Color Contrast)CC + (Superpixels Straddling)SS (step 1). They use c to score each window in l (step 2). They then run the non-maxima suppression. This results in a set ρ_s of promising windows

Algorithm 1: Using objectness for class-specific detectors.

Input: F, D, c
Ouput: Det
Step 1: $l = \{w_1, ..., w_F\}, w_i \rightarrow D, \forall_i$
Step 2: $l_s = \{(w_1, sw_1), ..., (w_F, sw_F)\}, sw_i = c(w_i), \forall_i$
Step 3: $\rho_s = NMS(l_s) = \{(w_{n1}, sw_{n1}), ..., (w_{np}, sw_{np})\}$
Step 4: $Ł = \{w_{n1}^{lm}, ..., w_{np}^{lm}\}, w_{nj}^{lm} = max(s_w)$
Step 5: $Det = NMS(Ł)$

(step 3). For every window $w_p \epsilon \rho_s$, they iteratively move to the local maximum of c in its neighborhood V_{wp}, resulting in window w_p^{lm} (step 4). Finally, they run NMS on the local maxima windows Ł and obtain detections Det (step 5). In order to use this algorithm one has to specify a window scoring function c, which is specific to a particular detector and object class, and a window neighborhood.

2) Selective Search for Object Recognition: J.R.R. Uijlings et al. [2] took a hierarchical grouping algorithm to form the basis of their selective search. Bottom-up grouping is a popular approach to segmentation, hence they adapted it for selective search. Because the process of grouping itself is hierarchical, they can naturally generate locations at all scales by continuing the grouping process until the whole image becomes a single region. This satisfies the condition of capturing all scales. As regions can yield richer information than pixels, they wanted to use region-based features whenever possible. To get a set of small starting regions which ideally do not span multiple objects, they used the fast method of Felzenszwalb and Huttenlocher [7], which found well-suited for such purpose. Their grouping procedure now works as follows. They first used [7] to create initial regions. Then they used a greedy algorithm to iteratively group regions together: First the similarities between all neighbouring regions are calculated. The two most similar regions are grouped together, and new similarities are calculated between the resulting region and its neighbours. The process of grouping the most similar regions is repeated until the whole image becomes a single region.

C. Architecture For Classifying Multi-Label Images

Our Final architecture for detecting multi-label images consists of (1) image segmentation and (2) convolutional neural network for detecting segmented images. The image segmentation techniques, objectness measures and selective search and the architecture for single label image detection has been described before. In the Figure 7 there is an overview of the architecture. At first we split the image with objectness measures and selective search. Then we get multiple split images. Note that there will be many split images. In the figure only a few are shown. Then we pass the split images through the convolutional neural network that classifies the label associated with that image. As there are many split images, there may be some labels that will be wrongly classified. We inspect these labels and their associative scores and finally conclude which of the labels are likely to be associated with the images.

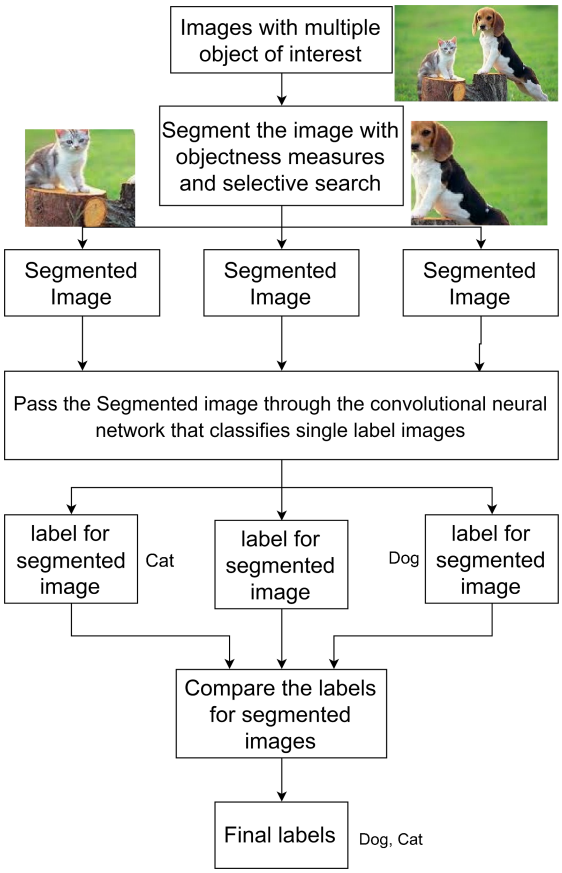


Fig. 7: Architecture of Convolutional Neural Network

We performed several techniques for inspecting the images. These are discussed in the experimental result section.

V. Experimental Result

A. Experimental Result for CIFAR-10

In the Table I there is the result for the testing images of CIFAR-10 dataset. Here, each row represents the testing label, each column represents the label produced by the network. For example the cell (1,1) represents that the testing label is airplane and the classified label is airplane, the cell (1,2) represents that the testing label is airplane and the classified label is automobile. So, each cell (i,i) where $[i = 1..10]$ represents the correctly classified labels. Each row consists of 1000 image labels. So the accuracy will be cell$(i, i)/1000$.

B. Experimental Result for Multi-Label Image classification

To give an overview of the process of inspecting the result of the segmented images, we will need the help of Table II. The table is for a single multi-label image. Each $image_i$ represents the segmented image from the multi-label image. Each row represents the class scores given by the network.

a) Selecting Top-1 Score: In this approach we select the top 1 score and its associating label from each segmented image $image_i$. Then we increase the frequency of labels for each top 1 score. After that we select the top 4 scoring labels. For example, in the Table II the top 1 label for $image_1$ would

TABLE I: CIFAR-10 Confusion Matrix

labels	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck	accuracy
airplane	915	4	17	19	3	1	0	2	27	12	91.50%
automobile	8	934	3	4	0	0	3	0	10	38	93.40%
bird	60	1	813	37	19	23	30	10	7	0	81.30%
cat	18	1	34	746	25	113	37	18	8	0	74.60%
deer	24	1	38	33	809	19	44	29	2	1	80.90%
dog	4	0	37	106	23	792	9	26	2	1	79.20%
frog	2	5	19	35	1	20	912	2	3	1	91.20%
horse	14	0	26	20	18	28	4	886	3	1	88.60%
ship	35	10	3	2	0	2	1	0	936	11	93.60%
truck	23	37	4	10	1	2	2	0	15	906	90.60%

be cat. So we increase the frequency of cat by 1. Similarly for $image_2$ again the score for label cat is highest. So frequency of cat will be increased by 1.

b) Selecting Top-2 Score: In this approach we select the top 2 scores and its associating label from each segmented image $image_i$. Then we increase the frequency of labels for each top 2 scores. After that we select the top 4 scoring labels. For example, in the Table II the top 2 labels for $image_1$ would be cat and deer. So the frequency of cat and deer will be increased by 1.

c) Selecting Total Score: In this approach we just add all scores of each $label_i$ for each segmented image $image_i$. Then we select the top 4 scoring labels. For example, every class score for every label will be added. So, the score for cat label will be the total score in the column cat. Similarly the score for bird will be the total score in the column bird.

d) Selecting Cumulative Percentage: Each $label_i$ of segmented image $image_i$ has the percentage $score_i / \sum_i score_i$. We select the top percentage from each segmented image $image_i$. We add these percentages for each associated labels and select the top 4 scoring labels. For example for $image_1$ the percentage of cat will be $score_{cat} / \sum_i score_i$.

The results for these four approaches for the sample image from Figure 8 is given at Table III and Table IV

These approaches were repeated in our selected multi label images. The results for Objectness Measure is given in Figure 9 and for Selective Search in Figure 10

Fig. 8: Sample Multi-Label Image

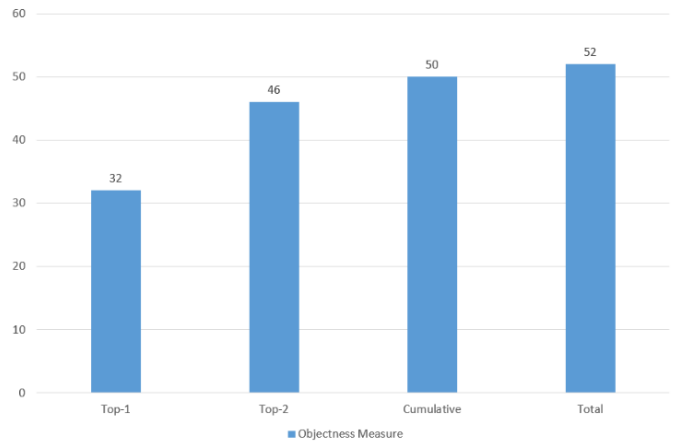


Fig. 9: Result for multi-label images for Objectness Measures

VI. Discussion

The main problem of classifying multi-label images are the training time required. If Convolutional Neural Network is used for classifying multiple labels, then there is a chance that the networks will be very complex. This will require a lot of time to train. Also, if it takes a lot of time to train, then tweaking the network and exploring promising functionalities will require a lot of time. But our Convolutional Neural Network can only classify Single Label images, then we use this network to classify the segmented images provided by Objectness Measures and Selective Search approach. The Convolutional Neural Network we provided is faster to train, so tweaking the network will provide quick result. We got about 57% accuracy using the Selective Search approach shown in II. So, we can expect that if we increase the capability of our network, this accuracy will be much higher.

VII. Future Work

The strength of our multi-label image classification architecture depends on convolutional neural network classifier and the method for segmenting images.So, be refining out convolutional neural network to include more labels by training CIFAR-100 and using better methods to segment the image, we can improve the performance of the network.As the image segmentation techniques give duplicate segmented images, so

TABLE II: Scores for the sample image

Images	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
Image-1	0.9217	0.0298	1.4209	46.4491	36.0700	2.5067	1.6066	0.5272	0.0714	0.3829
Image-2	1.4132	0.0659	7.7162	62.3326	1.1682	2.3488	1.5611	0.2317	0.2492	0.2310
Image-3	0.4459	0.0162	58.4923	246.7195	1.1131	8.1225	2.1280	0.4308	0.0539	0.1051
Image-4	1.0044	0.0395	1.7086	51.9305	6.6794	3.5150	0.9439	1.1725	0.1186	0.3051
Image-5	0.3337	0.0217	5.7818	347.2959	1.9758	11.4527	1.7994	0.7569	0.0565	0.1809
Image-6	1.3842	0.0419	8.2234	40.1665	3.7619	1.3729	2.1954	0.3672	0.1542	0.2732
Image-7	1.6438	0.0423	19.3501	62.6969	1.6350	2.1218	1.9803	0.3463	0.1220	0.1529
Image-8	0.7810	0.0442	2.1775	35.6856	11.8591	1.5890	3.8482	0.3406	0.1322	0.3772
Image-9	1.5966	0.0967	2.3003	66.9341	0.7218	8.9363	0.3764	1.1311	0.2680	0.1924
Image-10	0.8212	0.0312	8.5449	62.8961	5.5377	2.0566	2.6111	0.3208	0.1013	0.2574

TABLE III: Result for the sample image for Objectness Measures

Labels	Label-1	Label-2	Label-3	Label-4
Top-1	cat	bird	airplane	dog
Top-2	cat	bird	deer	dog
Cumulative	cat	deer	bird	dog
Total Score	cat	bird	deer	dog

TABLE IV: Result for the sample image for Selective Search

Labels	Label-1	Label-2	Label-3	Label-4
Top-1	bird	cat	frog	dog
Top-2	cat	bird	dog	frog
Cumulative	cat	frog	bird	dog
Total Score	cat	frog	bird	dog

to sort out all the unique segmented images can be a nice challenge to achieve.

VIII. Conclusion

We have presented an easy and simple approach to classify multi-label images using a trained CNN classifier with objectness measure and selective search of an image. It gave us the result in quick time. Our training time for CNN network is fast. Segmenting the images is also fast. Our technique is a stepping stone to this approach. It is giving a good threshold result. In the future, this model can be enhanced and cope with larger dataset. So it will be interesting to see how advancement is made in this approach.

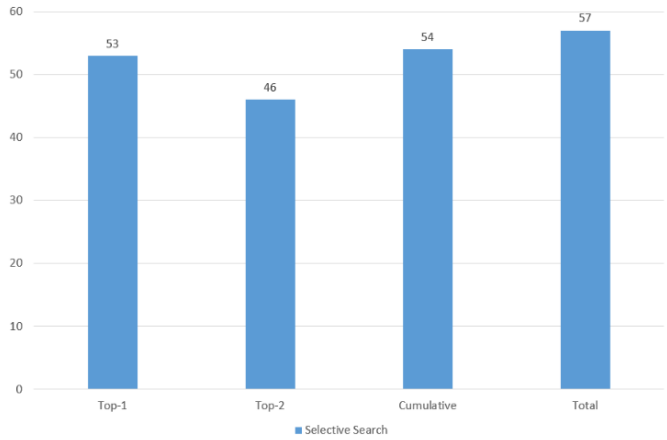


Fig. 10: Result for multi-label images for Selective Search

References

[1] Alexe, B., Deselares, T. and Ferrari. Measuring the objectness of image windows. V. PAMI 2012.

[2] R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, Arnold W. M. Smeulders. Selective Search for Object Recognition, Jasper International Journal of Computer Vision, Volume 104 (2), page 154-171, 2013.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In Neural Information Processing Systems, pages 11061114, 2012.

[4] Dataset of CIFAR-10 https://www.cs.toronto.edu/ kriz/cifar.html.

[5] Y. Gong, Y. Jia, T. K. leung, A. Toshev, and S. Ioffe. deep convolutional ranking for multi label image annotation. In International Conference on Learning Representations, 2014.

[6] Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

[7] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient Graph Based Image Segmentation. IJCV, 59:167181, 2004.