	MIPAL_SNU
Team leader name	Sungheon Park
Team leader address, phone number and email	Address : 864-1, Iui-dong, Yeongtong-gu, Suwon-si, Gyeonggi-do, Korea (443-270) Phone : +82-31-888-9579 Email : sungheonpark@snu.ac.kr
Rest of team members	Nojun Kwak
Team website URL (if any)	http://mipal.snu.ac.kr

General method description	Our method is mainly based on the deep convolutional neural network. Rather than using the whole image to train the network, our approach extract meaningful patches from each image. Similar to [3], we used the method of [1] to extract regions from images. Then, the patches are trained using deep convolutional neural network(CNN) which has 3 convolutional layers and pooling layers, and 2 fully-connected networks. Caffe framework[2] is used to train CNN. After training, class probabilities are calculated for every image patch from test image. Then, class probabilities of the test image is determined as a mean of the probabilities of all patches after the patch thresholding step.
References	[1] Uijlings, J. R. R. and van de Sande, K. E. A. and Gevers, T. and Smeulders, A. W. M., "Selective Search for Object Recognition", IJCV, 2013 [2] Jia, Yangqing and Shelhamer, Evan and Donahue, Jeff and Karayev, Sergey and Long, Jonathan and Girshick, Ross and Guadarrama, Sergio and Darrell, Trevor, "Caffe: Convolutional Architecture for Fast Feature Embedding", arXiv, 2014

<table>
<tr><td></td><td>[3] R. Girshick, J. Donahue, T. Darrell, J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”, CVPR, 2014</td></tr>
<tr><td>Describe features used or data representation mod el (if any)</td><td>Our CNN consists of 7 layers including input layer. The detailed structure is as follows(height*width*channels)
<table>
<tr><th>Layer</th><th>Input size</th><th>Kernel size</th><th>Output size</th></tr>
<tr><td>Input</td><td>129*129*3</td><td>-</td><td>-</td></tr>
<tr><td>Convolution1</td><td>129*129*3</td><td>5*5, stride 2</td><td>63*63*96</td></tr>
<tr><td>Pooling 1</td><td>63*63*96</td><td>2*2</td><td>31*31*96</td></tr>
<tr><td>Convolution2</td><td>31*31*96</td><td>3*3, padding1</td><td>31*31*128</td></tr>
<tr><td>Pooling2</td><td>31*31*128</td><td>2*2</td><td>15*15*128</td></tr>
<tr><td>Convolution3</td><td>15*15*128</td><td>3*3, padding1</td><td>15*15*128</td></tr>
<tr><td>Pooling3</td><td>15*15*128</td><td>2*2</td><td>7*7*128</td></tr>
<tr><td>Fully-connected1</td><td>7*7*128</td><td>-</td><td>2048</td></tr>
<tr><td>Fully-connected2</td><td>2048</td><td>-</td><td>2048</td></tr>
<tr><td>Softmax</td><td>2048</td><td>-</td><td>50</td></tr>
</table></td></tr>
<tr><td>Dimensionality reduction technique applied (if any)</td><td>Pooling step in the CNN</td></tr>
</table>

Classifier or method used to train and validate your results (if any)	For our CNN, the class probability is determined by softmax function, and cross entropy is used for error measure.
Large scale strategy (if any)	It is hard to train our network with about 7000 images. Therefore, extracting meaningful region using selective search[1] helps increasing the size of the training data in order to train the network. Approximately 200-400 patches are extracted from one image. The number of patches used for training was about 2100000.

Compositional model used (scene context representation), i.e. pictorial structure (if any)	We didn't explore any relationship between the patches. This may be considered as a future work.
Other technique/strategy used not included in previous items (if any)	When extracting image patches, we excluded patches that are too small or have too short patches. Also, after the patch classification, we excluded the patches that has high entropy(>2.5) since those patches don't have discriminative information. Each image patch is resized to 133*133 and randomly crop 129*129 region to give an input to CNN.
Method complexity analysis	It is hard to analyze the time complexity of CNN. The running time is reported in the page 7.

Results of the comparison to other approaches (if any)	With the patch probabilities, we take several approaches to evaluate the probabilities of one query image. We found that simply averaging over all patches shows good result. (mAP on validation dataset) Argmax of patches : 0.203 Weighted average where weight is proportional to the area of patches : 0.674 Mean of patches: 0.676 Mean of patches after thresholding by entropy : 0.683
Novelty degree of the solution and if is has been previously published	This method has not been published before. Though our approach is similar to [3], we used multiple patches for classify one image unlike [3] which is used for object detection.

Human effort required for implementation, training and validation?	We adjusted the threshold of entropy by calculating mAP on the validation set.
Training/testing expended time?	- Hardware : Intel Xeon CPU, 24 GB RAM, GTX TITAN Black GPU - For training (Using GPU) Extracting patches : 10-11hours, Training CNN: 16hours - For testing (Using only CPU) Extracting patches : 5-6hours, Testing CNN: 3hours
General comments and impressions of the challenge	