COST EFFICIENT SUBCATEGORY-AWARE CNN FOR OBJECT DETECTION

*Tingfeng Li, Xu Zhao**

Key Laboratory of System Control and Information Processing MOE
Department of Automation, Shanghai Jiao Tong University

ABSTRACT

In this paper, we propose an accurate and cost efficient deep CNN network for object detection. In contrast to the previous region-based methods like Sub-CNN [1], our detector is almost fully convolutional so that the computation cost can be reduced efficiently. By introducing position-sensitive score maps and exploiting subcategory information, our method is less time consuming while maintaining competitive performance on detecting objects with various scales. In addition, we remove image pyramid used in Sub-CNN to achieve further acceleration. The experimental results show that our approach is 1.3 times faster than Sub-CNN with only 14% number of parameters and archives comparable results on the challenging KITTI dataset. Compared with the state-of-the-art methods for object detection, our approach provides an efficient solution that takes into account both accuracy and speed.

***Index Terms*—** Subcategory, Convolutional Neural Network, Object Detection

1. INTRODUCTION

In recent years, Convolutional Neural Network (CNN) has shown great potential in solving a large number of computer vision problems, especially for visual object detection.

For CNN based object detection, R-CNN [2] is one of the earliest success attempts to apply CNN in object detection. With the paradigm of proposal and detection network, this method achieves high accuracy yet limited in computational efficiency. Fast R-CNN [3] and Faster RCNN [4] follow this paradigm but provide a better trade-off between accuracy and speed. To generate region proposals, traditional selective search [5] is used in [3] while a region proposal network is proposed in [4]. These methods attain comparable performance on commonly used benchmarks such as PASCAL VOC [6] and ImageNet [7]. Nevertheless, when it comes to the KITTI benchmark [8] for autonomous driving, where objects are usually small with regularly appeared occlusion and truncation, they perform relatively poor. Sub-CNN [1] solves the variation by introducing image pyramid and subcategory. This method usually produces the most accurate detection but tends to be slow since the image pyramid in CNN occupies many computational resources.

Another typical paradigm, such as YOLO [9] and SSD [10], directly predict location and object category by regression, which combine classification and localization into one aggregated network while exploit features in the whole image. This kind of methods are fast enough to be applied in practical applications, however, are limited in excavating local information to detect small objects.

To make full use of local and global information, in this paper, we choose to use proposal based method. To reduce computational cost, we design a nearly fully-connected-layer-free detection network. This is motivated by the recent state-of-the-art networks such as GoogleNets [11, 12] and ResNets [13], which are fully convolutional and are demonstrated to be very fast. In contrast, the traditional backbones like AlexNet [14] and VGG Nets [15], usually include a convolutional sub-network ending with several fully-connected (*fc*) layers taking all neurons in the previous layer into computation thus is slow.

There are two main basic strategies to achieve multi-scale detection. The first is to build an image pyramid and learn a single classifier. This tends to be very costly. Sub-CNN adopts the strategy. The second is to input single scale image and apply multiple classifiers. In contrast, this strategy avoids the repeated computation thus is more efficient. In the first stage, we principally aim at generating proposals with diverse scales and aspect ratios in possible locations. Implementing an image pyramid with a simple backbone is acceptable. Therefore, we adopt the same scheme as in [1]. On the second stage, the detection network is expected to learn a classifier and a regressor to accurately discern the category of objects and refine locations. In Sub-CNN, this is accomplished, again, by employing an image pyramid. Since more sophisticated backbone is needed in this phase, image pyramid can not be fed into the memory. In this paper, we use single scale image. By substituting fully connected layers with effective position sensitive RoI pooling, we achieve satisfactory performance as well as high computation speed.

Moreover, to recognize the same object with variant appearances, shapes and profiles, we utilize subcategory information as in [1]. Subcategories are generated automatically by clustering according to the properties or attributes such as

This research has been supported by the funding from NSFC (61673269, 61375019, 61273285). * indicates corresponding author.

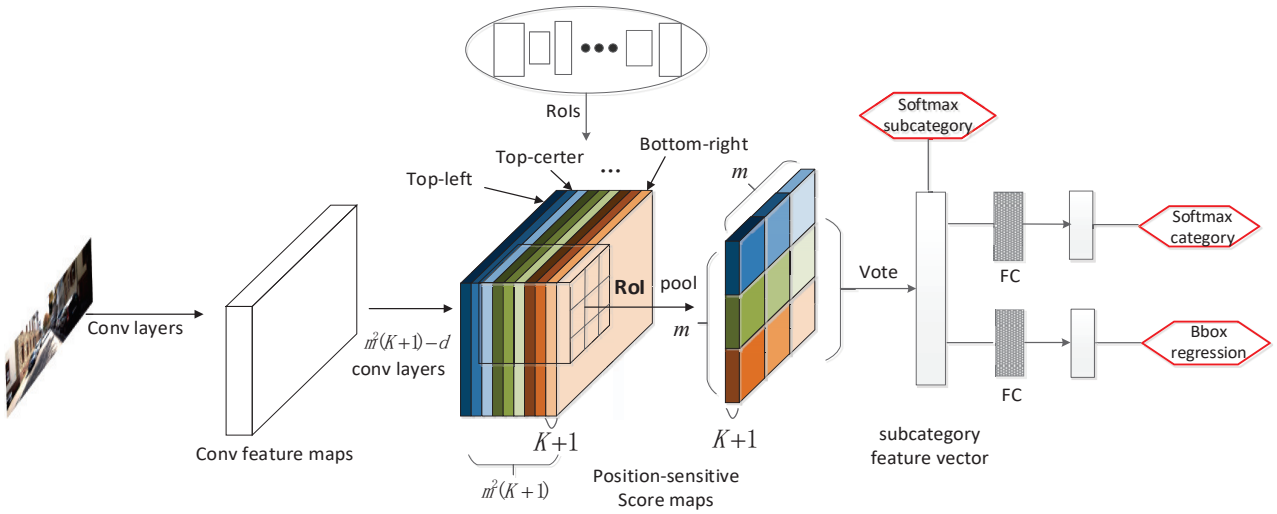

Fig. 1: Architecture of our object detection network. Red arrows indicate the route of derivatives in back-propagation training.

2D appearance, 3D pose or 3D shape. In region proposal network, subcategory facilitates the network to focus on certain locations where objects occur. In detection stage, we use a detection network to exploit subcategory information in proposals to classify their class and further refine locations.

Our approach is comprehensively evaluated on KITTI dataset. Without image pyramid and *fc* layers, our approach is 1.3 times faster than Sub-CNN with only 14% number of parameters and archives comparable accuracy.

2. PROPOSED APPROACH

2.1. The Architecture of Our Networks

Subcategory-aware RPN (Region Proposal Network). We use the RPN adopting commonly used image pyramid proposed in [1] to generate diverse scales of candidate proposals. First, it generates convolutional feature maps for each scale of the pyramid, which is a multi-dimensional array of size $C \times H \times W$, with C channels, H rows and W columns. Then a feature extrapolating layer follows to approximately produce a fine-grained scales of pyramid. Then a specially designed *conv* layer is applied for subcategory classification, where each filter in the *conv* layer corresponds to an object subcategory. These filters are trained to respond on specific locations, scales and subcategory of objects during training.

The *subcategory conv* layer consists of $K+1$ convolutional filters with K subcategories. Then a heatmap of size $K \times H \times W$ is generated by cross-channel max pooling. Each value in the heat map indicates the probability of an object presenting in the corresponding location, scale and subcategory. The RoI (Regions of Interest) generating layer then proposes potential candidate boxes according to responses in the heat map of each scale. In testing, wherever the response in a location is larger than a threshold, we use it to generate ROIs. In the end, we adopt RoI pooling, softmax and bounding box regression as in Fast R-CNN. The difference lies in that a subcategory-aware convolution substitutes the *fc* layers before softmax. This layer shares weights with the one mentioned before.

Position-sensitive subcategory-aware detection network. The architecture of the detection network is illustrated in Fig. 1. Given RoIs, the detection task can be reduced to classify them correctly. Therefore, in the second stage, we primarily aim at learning a classifier to determine whether a RoI output from the first stage encircle an object or not. Motivated by the traditional method as DPM [16], we utilize subcategory information to promote category classification like [1]. In KITTI, we adopt 3D Voxel Pattern (3DVP [17], including 3D pose, segmentation boundary and occluded regions) for Car and 2D appearance for Pedestrian and Cyclist. Unlike [17] and [4], both of which append an RoI pooling layer with several *fc* layers in the end thus drastically slow down the forward propagation, motivated by recent fully convolutional networks, we remove most of the *fc* layers, replacing with cost efficient *conv* layer and average pooling. The incarnation of the network is based on VGG16. After conv5_3, as in R-FCN, we add a convolutional layer to produce a pile of position-sensitve score maps encoding scores for each subcategory. Suppose one RoI is divided into m^2 bins, then this layer has $m^2 \times (K+1)$ channels with $(K+1)$ denoting the number of subcategories and background. Following is a position-sensitive RoI pooling layer which pools features from last *conv* layer over corresponding channel. This operation can be defined as:

$$r_k(i,j) = \sum_{(x,y) \in bin(i,j)} z_{i,j,k}(x + x_0, y + y_0)/n_{i,j,k} \quad (1)$$

where $r_k(i,j)$ denotes the reponse of the (i,j)-th bin for the k-th subcategory, (x_0, y_0) denotes the top-left corner of an

RoI, $z_{i,j,k}$ is the score map area corresponding to the (i,j)-th bin, $n_{i,j,k}$ is the number of pixels in this bin. Then vote through averaging operation for each subcategory to produce a $(k+1)$-dimensional *subcategory feature vector*, which is able to directly predict subcategory probability of candidate boxes. A softmax layer of category and a bounding box regressor is appended to predict based on this vector.

2.2. Parameters of Detection Network

In the original Sub-CNN, since it adopts VGG16 as backbone, most of parameters are in the fully connected layers. Both the first and second *fc* layer has 4096 neurons, possessing 102M and 17M parameters respectively. And the third one has 174 neurons for KITTI benchmark. While in our detection network, all the three *fc* layers are removed, substituting with one additional *conv* layer following position-sensitive ROI pooling layer and an average pooling layer, which only has 4M parameters in total. Consequently, the number of parameters in detection network has been drastically reduced to 14% comparing to the original detection network. The detailed comparison of parameters is shown in Fig. 2.

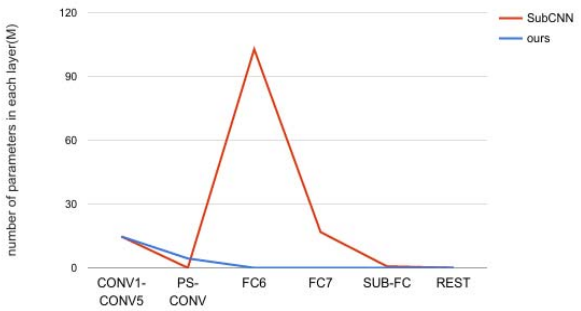

Fig. 2: Comparison between Sub-CNN and ours on number of parameters in detection network. "CONV1-CONV5": all *conv* layers in VGG16. "PS-CONV": position-sensitive *conv* layer in our detection network. "FC6", "FC7" and "SUB-FC": the first three *fc* layers in Sub-CNN. "REST": the two *fc* layers before softmax and bbox regression.

2.3. Loss Function

We train both of our networks with multi-task loss. For RPN, the loss is for subcategory classification and bounding box regression:

$$L(p,p^*,p'^*,t,t^*) = L_{subcls}(p,p^*) + \lambda[p'^*>0]L_{loc}(t,t^*) \quad (2)$$

where, p is a probability distribution $p=(p_0,p_1,...,p_K)$ for $K+1$ subcategories, p^* is the truth subcategory label, t is a vector representing the prediction of 4 parameterized coordinates of a bounding box and t^* is the bounding box regression target for class p'^* ($[p'^*>0]$ indicates that only foreground target is counted).

Table 1: Comparison of detection network using different number of bins m on KITTI validation set.

m	Object Detection (AP)								
	Car			Pedestrian			Cyclist		
	E	M	H	E	M	H	E	M	H
3	90.34	78.55	64.33	87.29	70.88	63.87	65.71	52.15	48.37
5	93.17	81.26	66.95	**88.50**	**71.60**	**64.49**	66.76	55.00	51.54
7	**94.03**	**84.05**	**70.26**	87.04	70.50	63.76	**70.07**	**56.20**	**52.50**

For detection network, the multi-task loss is composed of joint object class classification, subcategory classification and bounding box regression:

$$\begin{aligned} L(p,p^*,p',p'^*,t,t^*) = L_{subcls}(p,p^*) \\ +\lambda_1 L_{cls}(p',p'^*) + \lambda_2[p'^*>0]L_{loc}(t,t^*) \end{aligned} \quad (3)$$

where, $p'=(p'_0,...,p'_{K'})$ is a probability distribution of $K'+1$ classes, t and t^* are the predicted vector and true bounding box regression target respectively.

3. EXPERIMENTAL VALIDATIONS

3.1. Dataset and Setting

We evaluate the proposed object detection framework on the KITTI detection benchmark and the PASCAL VOC 2007 [19] dataset. Since the groundtruth annotations of the KITTI test set are not available, as in Sub-CNN, we split the KITTI training images into a train set consisting 3682 images and validation set containing 3799 images. The proposed object detection approach is implemented on Intel(R) Xeon(R) 2.10 GHz CPU with 32GB RAM, NVIDIA TITAN X GPU. In RPN, we use 5 scales and 7 aspect ratios for KITTI, 4 scales and 5 aspect ratios in PASCAL VOC. As in Faster RCNN, we use image-centric sampling, each SGD mini-batch is constructed from a single image. A mini-batch is expected to have 64 positive RoIs and 64 negtive RoIs selected by hard example mining. For detectin network, a mini-batch is constructed from 2 images with size 128 with 1:3 ratio of positive and negative RoIs. We use a learning rate of 0.001 for the first 30k mini-batches and 0.0001 for the rest 10k mini-batches on both datasets. We use a weight decay of 0.0005 and a momentum of 0.9 [14]. Our implementation uses Caffe [20].

3.2. Evaluation on Improved Components

For detection network, on KITTI, our detection framework is evaluated at three levels of difficulty, i.e., easy, moderate, hard, as suggested by [8]. We use Average Precision (AP) to measure the detection preformance.

Scales. In Table 2, we evaluate our detection network under different sizes of input image. Since image pyramid

Table 2: Comparison of detection network using different sizes of input image on KITTI validation set. $Scale$ is the resize factor. m is set to 7 and average pooling is used to vote. "E", "M", "H" stands for Easy, Moderate and Hard respectively.

$Scale$	Object Detection (AP)									Test Time (s/img)
	Car			Pedestrian			Cyclist			
	E	M	H	E	M	H	E	M	H	
1.0	89.86	74.08	60.97	86.18	68.51	61.61	50.88	41.29	38.89	1.211
1.5	93.89	82.03	68.54	83.49	68.35	61.85	62.07	48.88	45.59	1.295
2.0	93.44	81.84	68.18	**87.04**	**70.50**	**63.76**	64.03	51.78	48.22	1.419
2.5	**94.03**	**84.05**	**70.26**	81.10	66.54	60.08	**70.07**	**56.20**	**52.50**	1.585
3.0	93.94	83.71	69.27	84.87	68.92	62.08	66.11	54.32	50.52	1.779

Table 3: Comparison of our detection network with Sub-CNN [1], Faster RCNN [4] and R-FCN [18] on KITTI validation set.

Methods	Object Detection (AP)									Test Time (s/img)
	Car			Pedestrian			Cyclist			
	E	M	H	E	M	H	E	M	H	
Sub-CNN [1]	**95.77**	**86.64**	**74.07**	86.43	69.95	64.03	**74.92**	**59.13**	**55.03**	2.093
Faster RCNN [4]	82.91	77.83	66.25	83.31	68.39	62.56	56.36	46.36	42.77	0.229
R-FCN [18]	92.10	73.67	59.88	76.73	59.28	51.81	45.84	37.55	35.32	0.183
ours	94.03	84.05	70.26	**88.50**	**71.60**	**64.49**	70.07	56.20	52.50	1.585

is removed, this setting is to compare the effect of image size. Where, $Scale$ is resize factor. For Car and Cyclist, $Scale = 2.5$ turns to be the best, while for Pedestrian, $Scales = 2.0$ is better. This is a little tricky because generally a smaller object should be detected more easily when it's enlarged more. We can explain it as pedestrian is smaller in size, over-enlargement may cause noise resulting from ambiguity. In addition, we can see that resizing image is a simple but effective way to boost performance. In comparison with methods using fixed size like SSD, resizing is more spontaneous as it preserves original aspect ratio of an image.

Number of bins. In Table 1, we compare different number of bins inside an RoI. Actually, the larger value the m takes, the more fine-grained RoI can be divided into. As can be seen, while $m = 7$ is best for Car and Cyclist, Pedestrian performs best when $m = 5$. This is because pedestrian is smaller comparing to other two classes so information in each bin will be not enough for classification when use larger m, which leads to worse performance.

3.3. Overall Comparisons with Other State-of-the-arts

We compare our approach with Faster RCNN [4], Sub-CNN [1] and R-FCN [18]. As presented in Table 3 , our detection network is 1.3 times faster than Sub-CNN. For the categories of Car and Cyclist, the AP of our method slightly drops comparing with Sub-CNN. However, our approach outperforms Sub-CNN for Pedestrian category. It indicates that position-sensitive scores maps are more effective for small objects. Compared with Faster RCNN and R-FCN, we provide a good trade-off between accuracy and speed with better accuracy though slower speed.

We also evaluate our method on PASCAL VOC 2007 detection benchmark. The results are shown in Table 4. As can bee seen, still our approach is faster than Sub-CNN but nearly no much improvement on AP can be achieved by using our approach. The possible reason is that region proposal on PASCAL VOC is relatively easy compared to KITTI so the advantages of our approach on handling small objects with occlusion and truncation can not be well utilized.

4. CONCLUSIONS

In this paper, we propose an accurate and cost efficient approach for object detection. Our detection network is nearly fully convolutional. Without image pyramid and removing most *fc* layers, substituting with one convolutional layer following position-sensitive RoI pooling layer, our method is 1.3 times faster with 14% number of parameters comparing to Sub-CNN though accuracy slightly drops. Our approach achieves comparable performance, exhibiting a better speed-accuracy trade-off.

Table 4: Comparisons between Faster RCNN [4], Sub-CNN [1], R-FCN [18] and our method on PASCAL VOC 2007 dataset.

	Test time (s/img)	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
[18]	0.148	**71.9**	**77.0**	79.9	**73.9**	**61.8**	**52.3**	78.0	**79.9**	83.8	**53.9**	**78.7**	62.8	**84.9**	**82.9**	**78.1**	**78.1**	**39.8**	**75.5**	**72.7**	76.5	**68.4**
[4]	0.198	69.9	70.0	**80.6**	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	**67.2**	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
[1]	1.863	68.5	70.2	80.5	69.5	60.3	47.0	**79.0**	78.7	**84.2**	48.5	73.9	63.0	82.7	80.6	76.0	70.2	38.2	62.4	67.7	77.7	60.5
ours	1.277	65.5	68.9	74.8	68.5	56.2	38.1	77.1	70.9	82.0	44.3	67.2	59.1	80.6	80.2	71.7	61.5	38.2	59.0	68.9	**82.7**	59.2

5. REFERENCES

[1] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[3] Ross Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[5] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[6] Mark Everingham, Luc Van Gool, Chris Williams, John Winn, and Andrew Zisserman, "The pascal visual object classes challenge 2012," 2011.

[7] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3354–3361.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 779–788.

[10] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg, "SSD: Single shot multibox detector," in *European Conference on Computer Vision*. Springer, 2016, pp. 21–37.

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[12] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[16] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[17] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese, "Data-driven 3d voxel patterns for object category recognition," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 1903–1911.

[18] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., pp. 379–387. Curran Associates, Inc., 2016.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," `http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html`.

[20] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.