Content-Based Image Recovery

Hong-Yu Zhou and Jianxin Wu

National Key Laboratory for Novel Software Technology
Nanjing University, China
zhouhy@lamda.nju.edu.cn
wujx2001@nju.edu.cn

Abstract. We propose an interesting challenge: recovering from aspect-ratio distorted images based on their contents. Given a distorted image, we want to construct a model to predict its original aspect ratio. Since this is a general task, we build a database on top of Pascal VOC datasets. On the base of recent deep convolutional neural networks (CNNs), we present a multi-scale architecture and construct a spatial pooling layer to overcome the problem. By utilizing the multi-level and spatial information, our approach surpasses other methods by a large margin. Towards a better understanding of this task, we also perform detailed studies on experimental results.

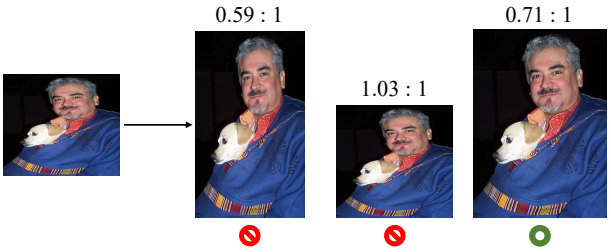
Keywords: Multi-scale CNN; spatial information; image classification; image aspect ratio recovery

1 Introduction

In recent years, CNNs have achieved great success in various image recognition tasks, e.g., object detection [5], semantic segmentation [6]. Thanks to these great achievements, deep learning is trying to bridge the gap between computers and humans. In image classification, we have surpassed human-level performance on the famous ImageNet challenge [7], and scene classification tasks [8] are no more difficult for deep CNNs [10] [9].

However, there are still doubts that high recognition performance means the computer has the same ability to understand image contents as we do. In this paper, we try to propose a new problem which is pretty normal for humans: *predicting the right aspect ratio of a single image*. Given a distorted image, human is able to give its suitable aspect ratio (Figure 1a). We argue that this ability relies on the recognition of the shapes of typical objects. For example, in Figure 1a, we can tell that the rightmost image is correct because we know what a person and a dog usually look like, which is a normal ability that all of us should have. Now suppose that we have tens of thousands images with distortion, how can we get back their original versions? Are our computers able to make predictions as accurate as we do? We will talk about this question in the rest of this paper.

In this paper, we try to give a thorough investigation of answers to the question above. Our contributions can be summarized as follows:



(a) Recover the original image

(b) Guess what is the aspect ratio?

Fig. 1: Predict the aspect ratio: (a) Considering a distorted image, we can recover the original image by predicting the right aspect ratio (the right picture). (b) Can you guess the original ratio? (answer: 1.33 : 1) Best viewed in color.

- For the first time, we propose to predict the original aspect ratio given a distorted image which can be regarded as an understanding towards image contents.
- We propose a multi-scale CNN architecture with spatial pooling layers to solve this problem and the proposed approach achieves better results over other traditional methods.
- Towards a better understanding of this problem, we build a new dataset with detailed annotations on top of existing datasets. We also perform ablation studies and statistical analysis on experimental results.

2 Related Work

There have been some works focusing on image transformation based on image contents [11] [12] [16]. He *et al.* [12] proposed a warping method that creates the perception of rotation and avoids cropping. They designed an optimization-based method that preserves the rotation of horizontal/vertical lines, maintains the completeness of the image content, and reduces the warping distortion. Hoiem *et al.* [11] presented a fully automatic method for creating a 3D model from a single photograph. The main insight is that instead of attempting to recover precise geometry, they statistically modeled geometric classes defined by their orientations in the scene. Li *et al.* [16] used a geodesic-preserving method

for content-aware image warping. However, these works only employed image geometry while ignoring semantic information. Different from these tasks, our goal is to recover the correct aspect ratio which needs better representations on semantic level.

Multi-scale CNNs have been developed to utilize multi-level features to get better performance, e.g., depth estimation [14], image classification [2], object detection [13] [15]. Eigen *et al.* [14] employed two deep network stacks: one that makes a coarse global prediction based on the entire image, and another that refines this prediction locally. Kim *et al.* [13] applied multi-scale hand-crafted features to car detection while Kong *et al.* [15] used multi-layers to extract CNN-based representations for object detection. Yang *et al.* [2] made a complete investigation into the details about multi-scale CNNs which somehow helps us design our deep model. However, the main difference is that our model uses *spatial pooling* to utilize spatial information apart from multi-level representations which make our model get better results over other multi-scale approaches.

3 Predicting the Aspect Ratio

We propose to use a multi-scale CNN to directly perform ratio regression. The overview of the model is shown in Figure 2. Note that we build model based on VGG-16 [4] but similar idea can be easily applied to other popular architectures (e.g., ResNet [17]). We argue that multi-scale and spatial information do help predict the original aspect ratio because they utilize low-level and mid-level representations.

3.1 Architecture

As shown in Figure 2, we mainly extract features from relu layers right before the pooling layers. The reason why we choose to use the last relu layer of each convolution block is that they can be regarded as the bottleneck of each block and have the ability to describe the whole block. Then we perform an operation called *Spatial Pooling* on these predetermined relu layers. The goal of this operation is to make use of spatial representations. More details about this block are given in Figure 3. Feature maps for specific scale are split into 2×2 regions and a max pooling layer is added on top of each region. A convolution layer and a relu layer are followed to extract feature vectors with fixed lengths. These feature vectors are then add together to produce the final feature representations.

We take L1 distance as the loss function,

$$\ell(\mathbf{X}, \mathbf{y}) = \sum_{i=1}^{N} |f_\theta(x_i) - y_i| \tag{1}$$

where $f_\theta(\cdot)$ represents the output of the network, x_i and y_i are input image and its label, $\mathbf{X}$ and $\mathbf{y}$ represent images and labels in the dataset, N is the number of training images. In our experiments, *we use the original aspect ratio of each image as its label.*

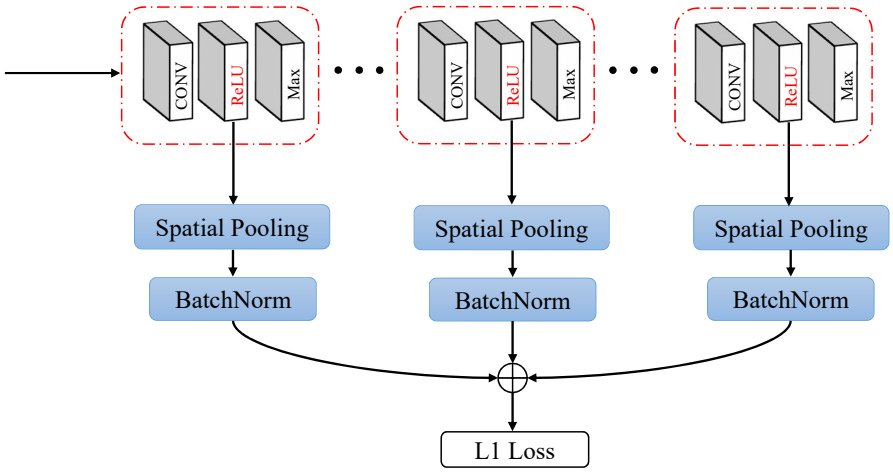


Fig. 2: An overview of our multi-scale model. We create a Spatial Pooling block which is able to utilize spatial information. We perform feature extraction on ReLU layers right before the pooling operations. Note that we use 4 scales in practice while this figure only shows 3 scales.

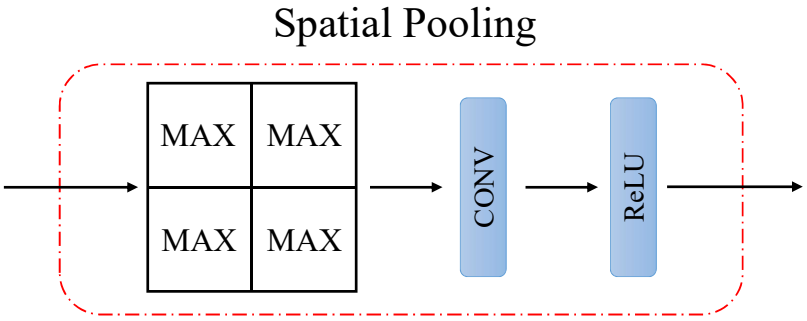


Fig. 3: On each feature map, we split the whole space into 2×2 regions and perform max pooling on each subregion. The pooling layer is then followed by a convolution layer and a relu layer to extract feature vectors. The number of regions can be a free choice and we also perform ablation experiments on different choices (in Table 3).

3.2 Implementation details

We implement the whole network using MatConvNet [19]. In our experiments, we build a 4-scale model (conv2, conv3, conv4 and conv5). In spatial pooling block, we split the whole image into 2×2 regions. We also use batch normalization on top of the loss function to make the training process easier. The learning rate is 10^{-2} and gradually reduces to 10^{-5} using logspace in 20 epochs. Details of the model architecture can be found in Table 1.

Table 1: Architecture details of the proposed CNN. You can go through the table from top to bottom which follows the implementation order. Note that parameters of batch normalization are the same as those in [18].

scale-1	**scale-2**	**scale-3**	**scale-4**	**filter size**
relu2_2	relu3_3	relu4_3	relu5_3	None
Spatial Max Pooling (2×2 regions)				None
Conv	Conv	Conv	Conv	2×2×{128, 256, 512, 512}×512
BatchNorm	BatchNorm	BatchNorm	BatchNorm	None
ReLU	ReLU	ReLU	ReLU	None
Sum				None
Conv				1×1×512×1
BatchNorm				None
L1 Distance				None

4 Experiments

4.1 Dataset

For the training and test data, we directly use the Pascal VOC 2007 and 2012 datasets [1] which are derived from the famous PASCAL Visual Object Classes (VOC) Challenge. Although it is possible to employ any other datasets instead, we choose the VOC series because of their detailed labels which can help us to analyze the experimental results. In practice, we use the test set of VOC2007 (4,952) as the test data while the rest images (16,551) are treated as training images. During the training process, we only perform left-right flip on each input image without any other data augmentation methods.

4.2 Baseline models

AlexNet, VGG-16 and ResNet-101. In AlexNet and VGG-16, we transform "fc8" from 1×1×4096×1000 to 1×1×4096×1 and replace dropout with

batch normalization. In ResNet-101, we directly change the number of final layer output from 1000 to 1. We also use batch normalization before the L1 loss to facilitate the training. The learning rate is 10^{-3} and gradually reduces to 10^{-5} using logspace in 20 epochs.

MS-VGGs. These models are all built based on VGG-16. We construct the main bodies of different variants following the same strategy stated in Table 1. It is worth noting that these variants are trained with the same learning rate as told in Sec 3.2.

4.3 Experimental results

The regression results of different models are reported in Table 2, and a few interesting points can be observed from it.

Table 2: Experimental results. We mainly perform experiments on AlexNet and VGG-based models. We use *MS-* to represent our multi-scale models. **SP**: spatial pooling (default 2×2); **Scales**: layers involved in multi-scale CNN; **Average Loss**: average L1 distance loss on test set (*the lower the better*).

Method	**SP**	**Scales**	**Average Loss**
AlexNet	w/o	w/o	0.27
VGG-16	w/o	w/o	0.18
ResNet-101	w/o	w/o	0.15
MS-VGG	w	conv5, 4	0.153
MS-VGG	w	conv5, 4, 3	0.127
MS-VGG	w	conv5, 4, 3, 2	0.112
MS-VGG	w	Conv5, 4, 3, 2, 1	0.129
MS-VGG	w/o	conv5, 4, 3, 2	0.126

MS-VGG with 2 scales performs as well as ResNet-101. MS-VGG with conv5, 4 gives 0.153 average loss which is only 0.03 higher than ResNet-101. This phenomenon tells us that multi-scale representations are able to help to recover from distorted images. Note that MS-VGG with 3 scales surpass ResNet-101 by 0.023 which implies that multi layers fusion might have a larger influence on results than simply increasing the depth of network.

More scales do not mean better performance. 3-scale MS-VGG exceed 2-scale model by 0.026 while 4-scale network achieves the best result. However, MS-VGG with full scales (5 scales) not only performs worse than 4-scale model but also loses out to 3-scale network. We can see that more layers might not lead to better results. We argue the reason might be that conv1 are too low to provide valuable representations.

Spatial pooling makes MS-VGG more powerful. MS-VGG with spatial pooling gets 0.014 points lower than that without this operation which suggests that spatial information might contribute to our task. By performing pooling on different regions, we are able to make a fusion of different positions in addition to different feature levels (multi-scale).

Table 3: Comparison between different spatial strategies. **Region size** tells us how to split the feature map.

Base model	**Region size**	**Average Loss**
MS-VGG	2×2	0.112
MS-VGG	3×3	0.105
MS-VGG	4×4	0.100
MS-VGG	5×5	0.102

We also compare different spatial pooling strategies and report the comparison results in Table 3. MS-VGG with 5×5 shows higher loss than 4×4 model which implies that more regions might not mean better performance.

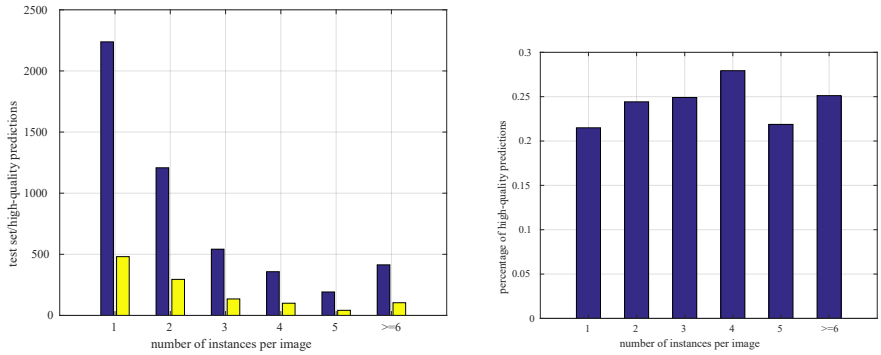
4.4 Results analysis

In Figure 4, we gives an analysis on the results produced by MS-VGG with 4 scales (2×2 regions). Our goal is to find if there exists correlation between the number of instances and high-quality predictions. *By saying high-quality predictions, we mean those images whose L1 loss are lower than 0.03.* In Figure 4a, we partition the test set (4,952 images) according to the number of instances in each image. *Note that* We can tell that images with few instances take the main part. As shown in Figure 4b, the percentage of high-quality outputs are almost the same among different types of images with specific number of instances. This might be a little surprising result which suggests that the number of objects has nothing to do with the difficulty of recovering the original images.

Table 4: The percentage of high-quality predictions in different partitions of original image aspect ratios.

0.2∼0.6	0.6∼1.0	1.0∼1.2	1.2∼1.4	1.4∼1.6	1.6∼1.8
0.0435	0.1366	0.1275	0.3346	0.1514	0.0122

Another question is: what is the relationship between the difficulty of recovery and the original image aspect ratio? To answer this question, we also make an



(a) Test set and experimental results (b) Percentage of high-quality predictions

Fig. 4: (a) We compare the number of instances and high-quality predictions. The blue and yellow histograms represent the number of images in test set and high-quality predictions, respectively.(b) We calculate the ratio between the number of test images and high-quality images.

investigation and show the results in Table 4, from which we can see that images whose aspect ratios are between 1.2 and 1.4 are the easiest ones to be recovered. Large aspect ratios (e.g., 1.6∼1.8) make it difficult to recover from distorted images.

We also show some predictions of our CNN model in Figure 5. Although the network still feels hard to recover images with large aspect ratios (row 1, 3, 6 in Figure 5), most of its outputs are at least acceptable and some of them are undistinguishable (row 2, 4, 5).

5 Conclusion

We propose to recover from aspect-ratio distorted images based on image contents. To solve this problem, we build a multi-scale architecture with spatial pooling that performs well on the recovery task. We perform complete ablation studies on details of the model architecture. Finally, we discuss the difficulty of predicting the original aspect ratio and try mining its relations with other factors, e.g., the number of instances.

References

1. Everingham, M., Eslami, S. M. A., Van Gool, L., Williams, C. K. I., Winn, J. and Zisserman, A.: The PASCAL Visual Object Classes Challenge: A Retrospective. In: International Journal of Computer Vision. 111(1), pp. 98–136 (2015)
2. Yang, S. F., Ramanan, D.: Multi-scale Recognition with DAG-CNNs. In: International Conference on Computer Vision (2015)

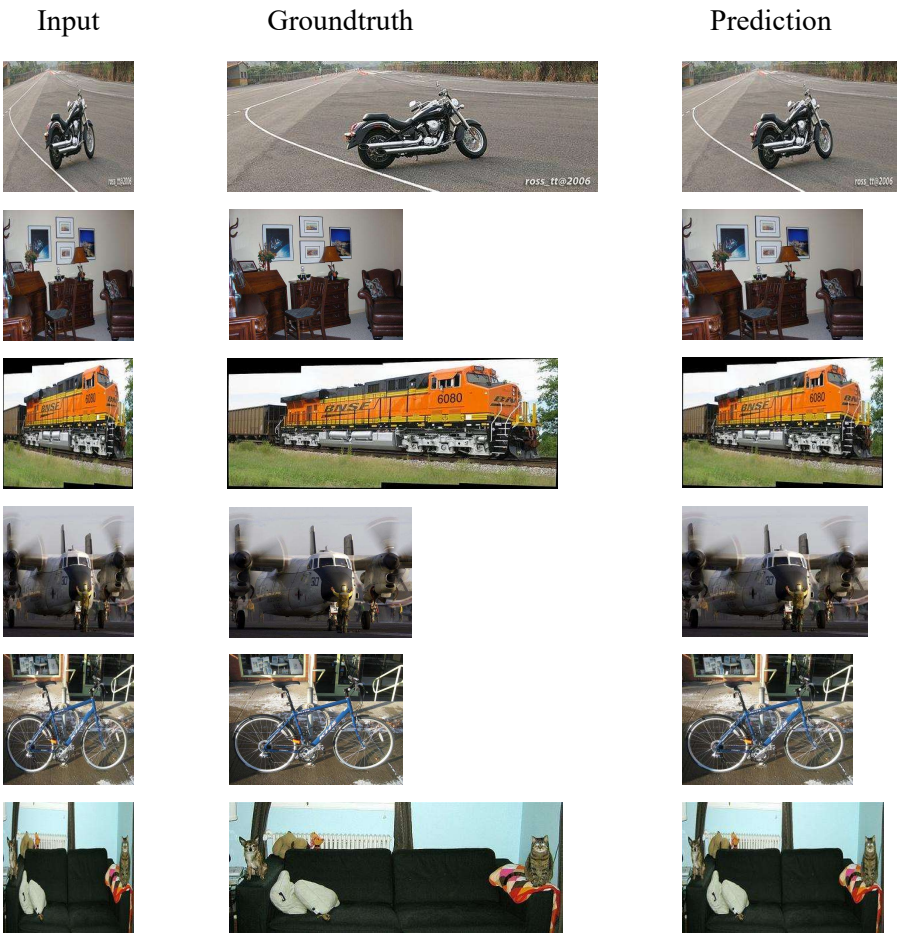


Fig. 5: Some examples of network predictions. The left column is the input image (all resized to 224×224), the middle column is the original image, the right column contains recovered images with predicted aspect ratios. *Note that heights of all images are fixed to the same length.*

3. Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks. In: Advances in neural information processing systems (2012)
4. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Technical report (2014)
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014)
6. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern

Recognition. pp. 3431–3440 (2015)
7. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. pp. 1026–1034 (2015)
8. Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3485–3492 (2010)
9. Xie, G. S., Zhang, X. Y., Yan, S., Liu, C. L.: Hybrid CNN and dictionary-based models for scene recognition and domain adaptation. In: IEEE Transactions on Circuits and Systems for Video Technology (2015)
10. Guo, S., Huang, W., Wang, L., Qiao, Y.: Locally supervised deep hybrid model for scene recognition. In: IEEE Transactions on Image Processing. 26(2), pp. 808–820 (2015)
11. Hoiem, D., Efros, A. A., Hebert, M.: Automatic photo pop-up. In: ACM transactions on graphics. 24(3), pp. 577–584 (2005)
12. He, K., Chang, H., Sun, J.: Content-aware rotation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 553–560 (2013)
13. Kim, J., Baek, J., Park, Y., Kim, E: New vehicle detection method with aspect ratio estimation for hypothesized windows. In: Sensors. 15(12), pp. 30927–30941 (2015)
14. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in neural information processing systems. pp. 2366–2374 (2014)
15. Kong, T., Yao, A., Chen, Y., Sun, F.: HyperNet: towards accurate region proposal generation and joint object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 845–853 (2016)
16. Li, D., He, K., Sun, J., Zhou, K.: A geodesic-preserving method for image warping. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 213–221 (2015)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
18. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. Technical report (2015)
19. Vedaldi, A. and Lenc, K.: Matconvnet: Convolutional neural networks for matlab. In: Proceedings of the 23rd ACM international conference on Multimedia. pp. 689–692 (2015)