IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 16, 2018, accepted September 24, 2018, date of publication September 28, 2018, date of current version October 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2872497

Constrained Optimization for Image Reshaping With Soft Conditions

CHEE SUN WON, (Member, IEEE)
Department of Electronic and Electrical Engineering, Dongguk University, Seoul 04620, South Korea
e-mail: cswon@dongguk.edu

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology under Grant NRF-2015R1D1A1A01057269 and Grant NRF-2018R1D1A1B07043542.

ABSTRACT Conventional image resizing problems demand hard conditions on size and aspect ratio, which must be met with no tolerance. In this paper, a generalized optimization framework is presented, which can handle soft conditions as well as the hard ones. The soft condition can be given by an allowable range of the image parameter, which is incorporated as an inequality condition in the constrained optimization framework. Given the soft constraints, the proposed framework seeks to find the set of image parameters that minimize the cost function. A constrained optimization via a linear programming framework is employed to manage a diverse combination of soft and hard conditions for the target image. The optimization is based on the image line, which optimally selects a set of image lines (columns and rows) to be deleted for size reduction in accordance with the cost function and the constraints. As a case study, the line-based optimal image resizing method based on the linear programming framework is applied for the pre-processing of VGG-19 convolutional neural network (CNN). Although the target input size is a hard condition of 224×224 for the VGG-19 CNN, the proposed optimization framework with a soft condition on the image size firstly finds an optimal near-square image with a tradeoff against the saliency level of image features. Then, the optimal near-square image is linearly scaled to the final image size to meet the hard condition.

INDEX TERMS Constrained optimization, convolutional neural network (CNN), image processing, linear programming.

I. INTRODUCTION

Image resizing methods may be categorized into two groups, namely the content-aware image resizing [1]–[5] and the feature-aware (or recognition-oriented) image resizing [6], [7]. In the content-aware image resizing methods, the main target for the size reduction (or expansion) is the pixels in the non-salient regions, preserving the salient visual information in the image. Since the content-aware image resizing is mainly for human vision, the resized image should be visually pleasing without noticeable distortions. The content-aware image resizing method has been evolved to further achieve the aesthetically pleasing display by learning the human gaze activities [8], [9]. On the other hand, the purpose of feature-aware (or recognition-oriented) image resizing approaches [6], [7] is to preserve local features for machine vision rather than to display for human vision. It has been shown that the feature-aware image resizing method can reduce the amount of image data efficiently, while preserving the image features sufficiently for visual search and retrieval [6], [7].

In the content-aware image resizing problem the size of the target image is a hard requirement for adapting the original image to the given display device. However, as the image resizing technique extends its scope to the feature-aware image resizing for machine vision, new requirements are added for the target images. That is, for the feature-aware image resizing approach, since the end-user of the resized image is a computer (machine) rather than a human being, the requirement to a fixed size can be relaxed. For example, if the purpose of image resizing is to reduce the amount of data for the wireless transmission to the server and is to be used for visual search, then the requirements such as fixed target size and fixed aspect ratio may not be necessary. Instead, a certain level of feature saliency needs to be imposed as a constraint to guarantee a minimal level of the feature matching and recognition performance. In this case, depending on the spatial distribution of local image features (e.g., the locations of key-points), the size and the aspect ratio of the resized images are allowed to vary and, thus, are given as soft requirements.

There are applications for the feature-aware image resizing that require the size of the target image to be fixed as a hard condition. For example, the CNN (Convolutional Neural Network) machines such as AlexNet [10] and VGGNet [11], which are being adopted for various applications [12]–[20] via the transfer learning, accept only a predetermined fixed-size input image. Specifically, the original images should be resized to 227 × 227 and 224 × 224 for AlexNet and VGGNet, respectively. Note that the original image size can be much bigger than the fixed-size CNN input, then the size reduction of the large image to relatively small one of 227 × 227 or 224 × 224 may cause substantial information loss and object distortion. In particular, since most CNN inputs are to be square, the object distortion due to the image resizing can be severe when the original image is thin or long with the aspect ratio much less or greater than 1. In fact, as reported in [21], the average side lengths of the popular 10 image datasets vary from 319.7 × 227.0 for Caltech-101 [22] to 1024.0 × 768.0 for Holiday datasets [23]. So, the image sizes are diverse and the aspect ratios can be far from 1 (i.e., not square). In such cases a substantial part of the object-of-interest (OOI) in the original image can be easily excluded in the resized image. To alleviate this problem an optimal balance between the linear scaling and the cropping may be useful, which forces to include the OOI as much as possible by the cropping but to avoid the shape distortion of the OOI by the linear scaling.

Various requirements can be imposed to the feature-aware image resizing problems, which makes it complex to find an optimal solution. This leads us to formulate the image resizing as an optimization problem with some constraints, where the constraints may include the aspect ratio, the saliency level, the amount of data, and the target image size, etc. In this paper, a generalized optimization solver for the image resizing with hard and soft constraints is proposed in the linear programming framework. The proposed image resizing framework uses straight horizontal and vertical lines [24], [25] instead of the pixel-wise curved seam as a basic unit for image resizing. Then, basically, the optimal image resizing is executed by pruning non-salient rows and columns in the image in accordance with the cost function and the constraints. Based on the line-based image resizing, it becomes a binary optimization problem, where a binary decision of line-deleting (binary 0) or line-keeping (binary 1) is made for each image line. Another merit besides the simplicity of the line-based image resizing is the size-recoverability to the original image from the resized one. That is, by sending the position information of the deleted lines to the receiver, they can be reconstructed from the undeleted neighboring image lines by interpolations [25].

The rest of the paper is organized as follows. In Section II, related works are reviewed from the perspective of various requirements in resizing images. Section III introduces the constrained optimization of image resizing under the linear programming framework. In Section IV, the image resizing is formulated in the quadratic programming framework. A case study for a fixed-size and near-square image resizing for the application of CNN input with experimental results is presented in section V. Finally, section VI draws a conclusion.

II. RELATED WORK

In [26], a mesh-based image resizing method was proposed to alleviate the distortions caused by the pixel-wise seam carving methods [1], [2], where the quads in the salient regions are resized uniformly to preserve the shape information, while those in the non-salient regions are the main target for deformations. So, instead of pixel-wise seam path, the quad in the mesh grid is the basic unit for the resizing and the deformations required for the image resizing are mainly imposed to the quads in the non-salient regions. However, as an extreme case, a quad can be contracted into a line or even a point. To solve this problem an axis-alignment condition is imposed to the non-uniform quad resizing as a constraint to the mesh grid optimization framework. Specifically, Panozzo et al. [4] have proposed a quadratic optimization framework with a constraint of a minimum size for each quad of the mesh grid. This quadratic optimization is adopted for the grid-warping based image resizing problems [6], [7], [27] for the sake of preserving local image features.

In [24] and [25] a straight horizontal or vertical line is used for the basic unit for image resizing rather than a pixel-wise curved seam or an axis-aligned quad. The straight image line can be viewed as a special case of the curved seam path in [1] and [2]. In fact, as observed in [25], the forward energy in [2] adopted for alleviating the distortion problem of the seam carving tends to make the seams straight and sparse. Also, it is computationally more attractive to choose a line (i.e., a column or a row) for the candidate of removal or duplication rather than finding a pixel-wise curved path as in the seam carving method. Another merit of the line-based image resizing is that, since the row or column numbers of the removed image lines can be efficiently stored and transmitted, the original image can be easily reconstructed by interpolating the removed rows and columns using the neighboring undeleted ones. This line-based image resizing is an optimization problem of choosing lines to be deleted and those to be remained in accordance with the saliency energy of the image [25].

As the image resizing technique evolves from the content-aware [1], [2], [26] for human vision into the feature-aware [6], [7] for machine vision, some requirements such as the aspect ratio to the target image can be relaxed because the resized image for the machine vision is not to be displayed. For some applications, however, the image reshaping for the machine vision also requires a fixed size and a fixed aspect ratio for the target image. For example, some CNN machines require a fixed-size input image (e.g., 224 × 224 or 227 × 227) [10], [11]. In this case the target image size and the aspect ratio should be imposed as hard constraints for the image retargeting. The state-of-the-art image resizing for CNN training employs a scale-jittering method [11], which first executes a linear scaling with a random scale

and then a random cropping to obtain the final fixed-size image. Thanks to the random execution of scaling and cropping, the resized images can take different parts with different scales in the original image and are used for the CNN training as a data augmentation technique. It has been shown in [11] that the scaling factor S chosen randomly from the range of $[256, 512]$ yields better results than the image resizing with a single scaling factor for training. However, this random scale jittering and the random cropping may cause substantial information loss and object distortion for images with much larger sizes than 224×224. This problem can be alleviated by adopting a multi-resolution CNN architecture [28], which consists of coarse resolution CNNs and fine resolution CNNs. Object information obtained from a large scale is described by the coarse resolution CNNs, while the fine resolution CNNs capture the detailed object information. Also, in [29], multiple subregions of the input image are selected as region proposals for CNN inputs. Note that each CNN with different resolution in [28] and the proposed regions in [29] still needs to be resized to a fixed one for CNN inputs. There are also CNN architectures that relax the fixed-size input requirements [30], [31]. In [30] a spatial pyramid pooling (SPP) layer was added on top of the last convolutional layer, where the front convolutional layers accept arbitrary size input images to generate the image feature map and the subsequent SPP layer pools the features and generates fixed-length outputs. However, without the pre-processing of the size reduction, the convolutional layers demand a lot of computationally expensive convolution operations for large images. Also, in order to fit the GPU memory, it will be computationally more attractive to reduce the image size [32]. Increasing popularity of CNN applications relying on the pre-trained CNNs with fixed-size images [12]–[19], including depth images [19] and key-frame in videos [20], also supports the necessity of the image resizing. The pre-trained CNN architectures with a fixed-size input image are also being adopted as a pre-processing or an initialization-processing network for the subsequent, more sophisticated, network architectures [33].

The requirements of the image reshaping for machine vision become diverse and any combination of the salience energy, the aspect ratio, and the target image size can be imposed as constraints for the optimization. Under this circumstance, it will be necessary to formulate the image reshaping as a constrained optimization problem. As mentioned already, the constrained optimizations in [4], [6], and [7] maximally preserve the local image features with the constraints of the fixed image size and the axis-alignment of the grids. In this paper a general framework of the line-based image resizing based on the linear programming is proposed, which facilitates different sets of constraints to the optimization framework. Note that the optimization frameworks for image resizing in [4], [6], and [26] are limited to a couple of hard constraints such as the target image size and the horizontal and vertical alignments of the quads in the grid. On the other hand, in this paper, conflicting requirements such as the level of image saliency, the aspect ratio, and the target image size can be considered simultaneously in the optimization framework as soft constraints with inequality bounds. This, for instance, allows us to have the size-reduced image such that the energy saliency is maintained at least 90% of the original one with a fixed target aspect ratio. Also, one can balance the level of energy saliency and the image size requirements by imposing a soft constraint such as a *near-square* size.

III. IMAGE RESIZING IN LINEAR PROGRAMMING FRAMEWORK

Image resizing can be done by selecting image lines (columns and rows) to be deleted (or to be duplicated), where the line selection criterion is based on the energy saliency (or a cost function) calculated for each image line. That is, given a saliency map $E = \{e(i,j) : 1 \le i \le M, 1 \le j \le N\}$ for the $M \times N$ original image I, the saliency for each image line can be obtained by simply accumulating the pixel energies along the image line

$$E_r(i) = \sum_{j=1}^{N} e(i,j), \quad i = 1, \cdots, M \tag{1}$$

$$E_c(j) = \sum_{i=1}^{M} e(i,j), \quad j = 1, \cdots, N \tag{2}$$

where $E_r(i)$ and $E_c(j)$ represent the saliency energies for the image-lines at row i and column j, respectively.

The line-based image resizing can be viewed as a binary classification problem such that each image line is to-be-remained or to-be-deleted. In [25] a MAP (maximum a posteriori) criterion with a Gibbs energy function [34] is employed for the optimization. In this paper, the binary decision is made by minimizing a cost function under some constraints. In particular, a binary linear programming approach is adopted to solve the optimization problem. That is, given the original image I of $M \times N$ with $M+N$ image lines, the goal is to make an optimal binary decision on each image line as either to-be-deleted or to-be-remained by using the integer (binary) linear programming optimization framework. Specifically, let us denote $1 \times (M+N)$ row vectors of f and s as the cost function to be minimized and the indicator of the binary decisions, respectively. Now, the cost function is given by the inner product between f and s. Here, $f = [f(1) \cdots f(M)\, f(M+1) \cdots f(M+N)]$, where the element $f(k)$ represents the cost for deleting the image-line at k. And the element of the binary indicator vector $s = [s(1) \cdots s(M)\, s(M+1) \cdots s(M+N)]$ takes $s(k) \in \{0, 1\}$, representing the binary decision of the line-keeping with $s(k) = 1$ or the line-deleting with $s(k) = 0$ at line k. Then, the optimal line selection can be made by following the linear programming framework with the cost function given by the inner product of f and s, and with some constraints

$$s^* = \arg\min_{s} f s^T \tag{3}$$

such that

$$As \leq b, \tag{4}$$
$$A_{eq}s = b_{eq}, \tag{5}$$
$$lb \leq s \leq ub, \tag{6}$$

where (4) and (5) are the soft and the hard constraints, respectively, and the associated matrices (or scalar) A, b, A_{eq}, and b_{eq} will be determined by the specific requirements for the optimization of (3). In the following subsections, some interesting combinations of specific hard and soft requirements expressed by A, b, A_{eq}, and b_{eq} will be presented. Note that, to yield binary values 0 and 1 for the element of s, the lower bound lb and the upper bound ub in (6) are set to zero vector and one vector with $M+N$ elements,[1] respectively.

A. CONSTRAINT ON IMAGE SIZE

The binary linear programming formulated in (3) – (6) is a general framework for the line-based image resizing problem and there is a freedom of setting the cost vector f in the objective function and the constraints in (4) – (6). For example, either the number of image lines to be deleted or the saliency energy of the image lines can be incorporated into the objective function.

In this subsection, the saliency energy is adopted as the cost function with the target image size as the constraint. That is, the size of $M \times N$ original image with $M+N$ image lines is to be changed to a target image-size of $M' \times N'$, where $M' \leq M$ and $N' \leq N$. If the aspect ratio of the target image is not fixed but the ratio of the number of target image lines to that of the original one is given by α ($\alpha < 1$) such that $M'+N' = \alpha \times (M+N)$, then this constraint can be expressed by $1 \times (M+N)$ vector of A_{eq} with $A_{eg} = [1\ 1 \cdots 1]$ and $b_{eq} = \alpha \times (M+N)$ in the hard equality condition of (5). The objective function $f = [f(1) \cdots f(M) f(M+1) \cdots f(M+N)]$ is a $1 \times (M+N)$ row vector and, to be used as a *cost* function, its element takes a negative value of the sum of the saliency energy for each image line as follows

$$f(k) = \begin{cases} -E_r(k), & \text{if } 1 \leq k \leq M \\ -E_c(k-M), & \text{if } M+1 \leq k \leq M+N \end{cases} \tag{7}$$

where $E_r(k)$ and $E_c(k)$ are the saliency energies for image lines as defined in (1) and (2). Then, the cost function, which is the inner product of f and s, is to be minimized under the hard constraint of the number of target image-lines to be $M'+N' = \alpha \times (M+N)$ as follows:

$$f = [-E_r(1) \cdots - E_r(M) \ - E_c(1) \cdots - E_c(N)] \tag{8}$$
$$s = [s(1) \cdots s(M)\ s(M+1) \cdots s(M+N)] \tag{9}$$
$$A_{eq} = [1 \ \cdots \ 1] \tag{10}$$
$$b_{eq} = \lceil \alpha \times (M+N) \rceil \tag{11}$$
$$lb = [0 \ \cdots \ 0]^T \tag{12}$$
$$ub = [1 \ \cdots \ 1]^T \tag{13}$$

[1]The linear programming parameters in (3)–(6) match MATLAB function *intlinprog*.

where each element of s takes a binary value such that $s(k) = 1$ for the line-keeping and $s(k) = 0$ for the line-deleting. In accordance with the above optimization framework the image lines with the large saliency energy now have small negative values as a cost due to the negative sign in f of (7) and are most likely to be remained with $s(k) = 1$. For example, if the parameter α, representing the ratio of the resized image to the original one, is set to $\alpha = 0.9$, then the optimal binary decision is to set $s(k) = 1$ for the smallest 90% of the elements in f with no consideration on the aspect ratio of the original image. Finally, the inequality constraint by $(M+N) \times 1$ column vectors of lb in (12) and ub in (13) is required for s to have binary values. Since the constraints of (12) and (13) are commonly required throughout the paper for the binary decision, except for the quad-based quadratic programming method in Section IV, it will be omitted in the rest of the linear programming frameworks.

B. CONSTRAINT ON ENERGY SALIENCY

We can set the number of image-lines as the cost function to be minimized so that the target image size is not fixed but varies from image to image. Instead, we can set the level of energy saliency as a soft constraint of (4) which is expressed as the lower bound of the total saliency energy for the image lines to be remained. Specifically, we set $A = [-E_r(1) \cdots - E_r(M) \ - E_c(1) \cdots - E_c(N)]$ and $b = -\beta \times (\sum_{i=1}^{M} E_r(i) + \sum_{j=1}^{N} E_c(j))$ for (4), where β is the ratio of the total saliency energy of the resized image to that of the original one. Again, $s(k) = 1$ is for line-keeping and $s(k) = 0$ for line-deleting. Then, by setting $f = [1 \cdots 1]$, the optimal solution of (3) selects the minimum number of image lines with the largest saliency energies for the line-keeping (i.e., $s(k) = 1$) such that the saliency-level constraint is satisfied. So, the linear programming optimization can be formulated by

$$f = [1 \ \cdots \ 1] \tag{14}$$
$$s = [s(1) \cdots s(M)\ s(M+1) \cdots s(M+N)] \tag{15}$$
$$A = [-E_r(1) \cdots - E_r(M) \ - E_c(1) \cdots - E_c(N)] \tag{16}$$
$$b = -\beta \times (\sum_{i=1}^{M} E_r(i) + \sum_{j=1}^{N} E_c(j)). \tag{17}$$

Fig. 1 shows the examples of the above two optimization frameworks with $\alpha = \beta = 0.9$, which corresponds to the optimization with 90% saliency-level constraint with (14)–(17) and the 90% line-keeping constraint for (8)–(13). Here, the saliency map is obtained by simply calculating the edge magnitude for each pixel and, certainly, can be replaced by any other saliency map dedicated to a specific application. Comparing with the original image in Fig. 1-(a) the main target of the line deletion is the monotonic region of the background. Note that the reduced image by the 90% saliency-level constraint in Fig. 1-(c) is smaller than that of the 90% line-keeping constraint in Fig. 1-(e), which can be reversed for other images with lots of complex and textured regions.

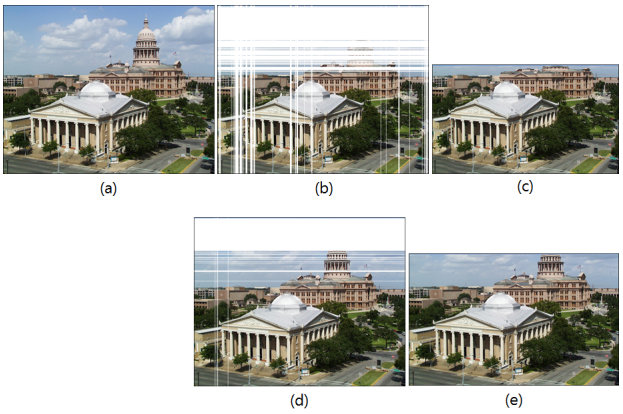

FIGURE 1. Size-reduction example with different cost functions and constraints: (a) Original image of 505 × 634, (b) Deleted lines in white by the 90% saliency-level constraint, (c) Reduced image of 329 × 560 without the white lines in (b), (d) Deleted lines in white by the 90% line-keeping constraint, (e) Reduced image of 395 × 630 without the white lines in (d). Original image from (http://live.ece.utexas.edu/research/quality/subjective.htm).

C. CONSTRAINT ON ASPECT RATIO

The optimal solutions in Fig. 1 can be also obtained by simply deleting the image-line with the lowest saliency energy repeatedly until the constraint is no loner satisfied. Therefore, in this case, we may not need an optimization solver. However, there is a soft requirement, such as the tolerable degree of a target aspect ratio, that makes the optimization more complex and needs the optimization solver.

To realize the linear programming optimization with the combined constraints of the target aspect ratio γ (i.e., $\gamma = \frac{width}{height}$) and the saliency-level we can set the linear programming framework as follows

$$f = [-E_r(1)\cdots - E_r(M) \; -E_c(1)\cdots - E_c(N)] \tag{18}$$

$$s = [s(1)\cdots s(M)\, s(M+1)\cdots s(M+N)] \tag{19}$$

$$A = \begin{bmatrix} \gamma & \cdots & \gamma & -1 & \cdots & -1 \\ -\gamma & \cdots & -\gamma & 1 & \cdots & 1 \end{bmatrix} \tag{20}$$

$$b = \begin{bmatrix} \Delta L \\ \Delta L \end{bmatrix} \tag{21}$$

where ΔL is the number of tolerable image lines for the given aspect ratio γ. Again, the binary indicator $s(k)$ takes $s(k) = 1$ for line-keeping and $s(k) = 0$ for line-deleting. The constraint with the above A and b matrices includes the upper and lower bounds for the aspect-ratio constraint, which limits the number of lines to be deviated from the exact aspect ratio γ. That is, $-\Delta L \leq \gamma \times M' - N' \leq \Delta L$, where M' and N' represent the sizes of the target image. The optimal solution should delete the image lines as much as possible, while keeping the ratio N'/M' as close to the given aspect ratio γ with the allowed margin ΔL. Fig.2 shows the resized images with $\gamma = 3/4$ and $\gamma = 4/3$ with $\Delta L = 1$.

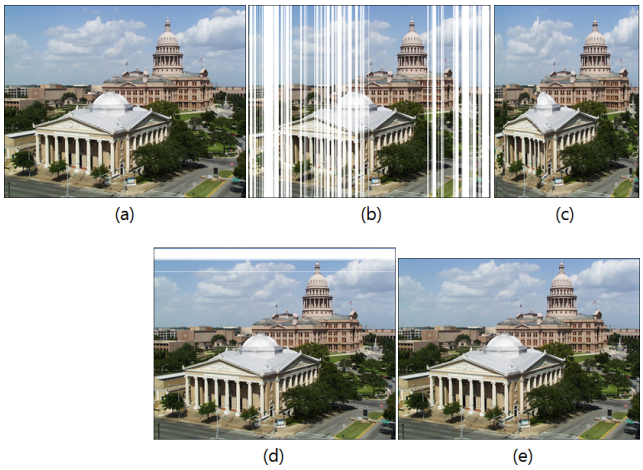

FIGURE 2. Image resizing under the aspect ratio constraints with $\Delta L = 1$: (a) Original image, (b) Deleted lines in white with the aspect ratio constraint $\gamma = 3/4 = 0.75$, (c) Reduced image with no white lines and $M' = 505$, $N' = 379$ ($379/505 \approx 0.75$), (d) Deleted lines in white with the aspect ratio constraint $\gamma = 4/3 \approx 1.33$, (e) Reduced image with no white lines and $M' = 476$, $N' = 634$ ($634/476 \approx 1.33$).

D. CONSTRAINT ON ADJUSTING LINE SPACING

As shown in Fig. 1 and 2 the lines to be deleted (i.e., the white lines in Fig. 1 and 2) tend to stick together, which may create a big discontinuity at the object boundary. This in turn generates new edges, resulting in artificially created salience regions in the resized image. To alleviate this problem we can add a constraint to split the consecutive would-be-deleted lines. This constraint can be added to (8) – (11) of the linear programming framework as follows

$$f = [-E_r(1)\cdots - E_r(M) - E_c(1)\cdots - E_c(N)] \tag{22}$$

$$s = [s(1)\cdots s(M)\, s(M+1)\cdots s(M+N)] \tag{23}$$

$$A = [1 \; -1 \; 1 \; -1 \cdots 1 \; -1] \tag{24}$$

$$b = -\zeta \times (1-\alpha) \times (M+N) \tag{25}$$

$$A_{eq} = [1 \; \cdots \; 1] \tag{26}$$

$$b_{eq} = \lceil \alpha \times (M+N) \rceil \tag{27}$$

Note that $M+N$ elements of A in (24) take alternating values of $+1$ and -1, which discourages the deleting image lines to be flocked together. Specifically, to meet the constraint of (24) and (25) and to minimize the saliency cost of (22) as well, some of the would-be-deleted lines with $s(k) = 0$ should be replaced by $s(k) = 1$ at the locations of -1 in (24). As a result, with the inequality constraint of (24) and (25), the optimization forces some of the consecutive lines with $s(k) = 0$ to have alternating labels of line-keeping ($s(k) = 1$) and line-deleting ($s(k) = 0$). In (25), ζ ($0 \leq \zeta \leq 1$) controls the total number of lines to split among all would-be-deleted lines of $(1-\alpha) \times (M+N)$, representing the amount of the forced splitting. As shown in Fig. 3, the number of split lines increases as ζ gets larger, where the dense white regions are split up into small white regions (notice the differences between Fig. 1-(b)(d) and Fig. 3-(a)(c)).

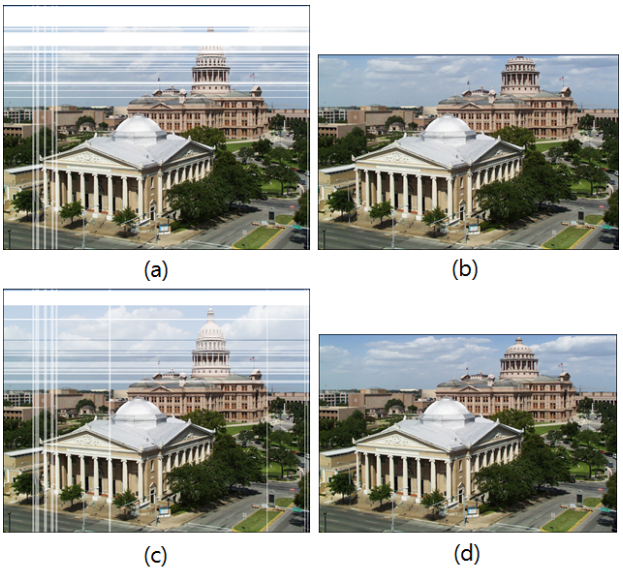

FIGURE 3. Image resizing by (22)–(27) with the dual constraints of the hard constraint of the number of line-keeping and the soft constraint of the line-spacing (white lines are to be deleted): (a) and (b) $\alpha = 0.9$ and $\zeta = 0.3$, (c) and (d) $\alpha = 0.9$ and $\zeta = 0.7$.

E. REQUIREMENTS ON COST FUNCTION

As in Section III-B the number of image lines can be used as a cost function, which is considered as a soft requirement. Similarly, image reshaping parameters can be incorporated into the cost function rather than the constraints. For example, the cost function in the linear programming framework of (8) – (11) can be modified as follows

$$\begin{aligned} f &= [-E_r(1)/M \cdots - E_r(M)/M \\ &\quad - E_c(1)/N \cdots - E_c(N)/N] \qquad (28) \\ s &= [s(1) \cdots s(M)\ s(M+1) \cdots s(M+N)] \qquad (29) \\ A_{eq} &= [1\ \cdots\ 1] \qquad (30) \\ b_{eq} &= \lceil \alpha \times (M+N) \rceil \qquad (31) \end{aligned}$$

In (28) the saliency energies $E_r(k)$ and $E_c(k)$ are scaled by the number of rows, M, and the number of columns, N, of the original image, respectively. Then, if $M = N$, there is no difference between (8) and (28). However, if $M > N$, then we have $-E_r(k)/M > -E_c(k)/N$ for $E_r(k) = E_c(k)$. This increases the chance of deleting image lines at the longer side (i.e., the rows with M lines) and the resized image tends to be square. So, if the original image is not square (i.e., the aspect ratio is greater or less than 1), the cost function in (28) encourages to choose more image lines from the longer side of the image for the line deletion. In Fig. 1-(e), the resized image has 395×630 with $\gamma = 0.63$, whereas the resized image in Fig. 4-(b) by the cost function of (28) has 387×524 with $\gamma = 0.74$.

The prior knowledge on the location of the region-of-interest (RoI) can be also incorporated into the cost function. Suppose that images to be resized have the RoI mostly around the center of the image, (i_c, j_c). Then, we can include the

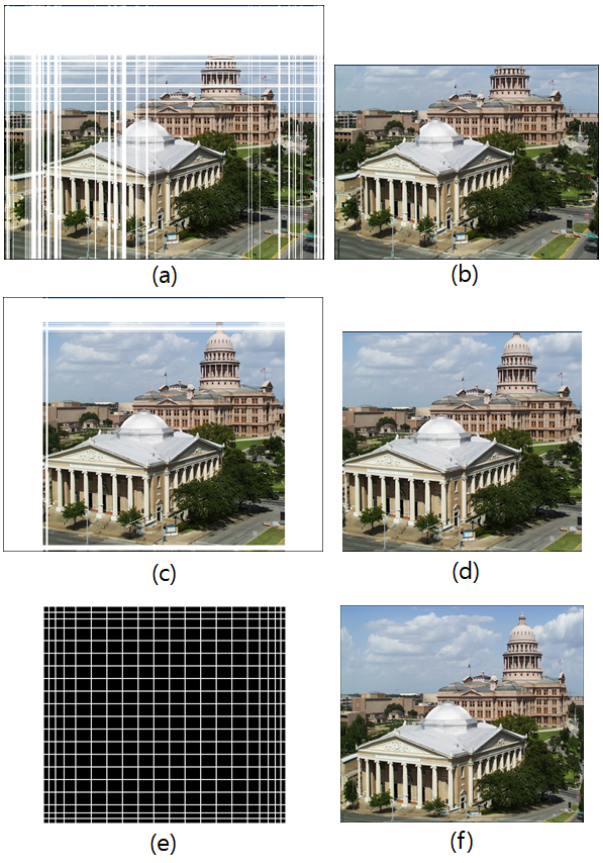

FIGURE 4. Comparisons for near-square image resizing with $\alpha = 0.8$, (a) and (b): Resized by (28) (size: 387×524 and $\gamma = 0.74$), (c) and (d): Resized by (32) (size: 436×475 and $\gamma = 0.92$), (e): Grid cells by the quadratic programming of (41) and (f): the resized image according to (e) (resized image has 430×481 and $\gamma = 0.89$).

RoI requirement into the cost function in (28) by multiplying the saliency energy $E_r(i)$ at the i^{th} row and the Gaussian weight of $G_r(i) = \exp(-(i - i_c)^2/2\sigma^2)$. Also, the saliency energy $E_c(j)$ at the j^{th} column is multiplied by its corresponding Gaussian weight $G_c(j) = \exp(-(j - j_c)^2/2\sigma^2)$ in f as follows

$$\begin{aligned} f &= [-E_r(1)G_r(1) \cdots - E_r(M)G_r(M) \\ &\quad - E_c(1)G_c(1) \cdots - E_c(N)G_c(N)] \qquad (32) \\ s &= [s(1) \cdots s(M)\ s(M+1) \cdots s(M+N)] \qquad (33) \\ A_{eq} &= [1\ \cdots\ 1] \qquad (34) \\ b_{eq} &= \lceil \alpha \times (M+N) \rceil \qquad (35) \end{aligned}$$

The Gaussian weighting factors $G_r(\ )$ and $G_c(\ )$ in (32) push the to-be-deleted lines away from (i_c, j_c), giving a cropping-like effect. As you can see in Fig. 4-(c), by setting $i_c = M/2$ and $j_c = N/2$, the lines to be deleted are pushed to the image border and more image lines at the central area of the original image are preserved. Another effect of the Gaussian weighting factors with $M \neq N$ is that the Gaussian weights decrease as the distance to (i_c, j_c) increases, forcing more image lines at the image border, especially at the longer side of the image, to be removed. Specifically, we can

set $\sigma = min(M, N)/5$ for both the Gaussian weights of G_r and G_c, encouraging more image lines at the longer side to be removed. This also reduces the gap between the horizontal and vertical sizes of the original image, yielding a near-square image.

IV. IMAGE RESIZING BY QUADRATIC PROGRAMMING FRAMEWORK

The quadratic programming optimization can be also employed for the image resizing along with most of the constraints used in the linear programming framework mentioned in the above sections. In [4], [6], [7], and [27], the uniform grid overlaid on the original image space is non-uniformly deformed for the axis-aligned image retargeting problem by the quadratic programming optimization. It can be reformulated as to determine the optimal number of image lines to be deleted for each horizontal (and vertical) stripe. That is, starting from equally spaced horizontal and vertical stripes with an identical width, the target widths of all stripes are optimally determined by the quadratic programming framework with constraints.

Overlaying a $U \times V$ grid over the original image of $M \times N$, each cell in the grid has the dimension of $\Delta U \times \Delta V$, where $\Delta U = \lfloor M/U \rfloor$ and $\Delta V = \lfloor N/V \rfloor$. Then, the original image I of $M \times N$ is trimmed to I^g of $M^g \times N^g$, where $M^g = U \times \Delta U$ and $N^g = V \times \Delta V$. Given the saliency map $E = \{e(i,j) : 1 \le i \le M^g, 1 \le j \le N^g\}$ for the image I^g the saliency energy for each grid-row (horizontal stripe) and grid-column (vertical stripe) is given by

$$E_r^g(u) = \frac{1}{\Delta U} \sum_{i=(u-1)\Delta U+1}^{u\Delta U} \sum_{j=1}^{N^g} e(i,j), \tag{36}$$

$$E_c^g(v) = \frac{1}{\Delta V} \sum_{i=1}^{M^g} \sum_{j=(v-1)\Delta V+1}^{v\Delta V} e(i,j), \tag{37}$$

where $E_r^g(u)$ and $E_c^g(v)$ represent the saliency energies for the grid at row $u = 1, \cdots, U$ and column $v = 1, \cdots, V$, respectively. Now, the quadratic optimization framework in [4] can be reformulated with the following quadratic cost function to optimally determine the cell sizes (i.e., the widths of stripes) in the grid in accordance with the saliency energies E_r^g and E_c^g

$$\arg\min_{s_u^r, s_v^c} \Big(\sum_{u=1}^{U} (\bar{E}_r^g(u) s_u^r)^2 + \sum_{v=1}^{V} (\bar{E}_c^g(v) s_v^c)^2 \Big) \tag{38}$$

$$= \arg\min_{s} s^T Q^T Q s \tag{39}$$

where s_u^r and s_v^c represent the row and the column sizes of the cell at the grid-row u and the grid-column v, respectively. Since the objective function is to be *minimized* in (38) and (39), the saliency energies E_u^r and E_v^c should be converted into the non-saliency energies by inversing them as $\bar{E}_u^r = 1/(1 + E_u^r)$ and $\bar{E}_v^c = 1/(1 + E_v^c)$. In (39), s is a $(U+V)\times 1$ column vector formed by $s = [s_1^r \cdots s_U^r s_1^c \cdots s_V^c]^T$ and Q is a $(U+V) \times (U+V)$ matrix such that

$$Q = \begin{bmatrix} \bar{E}_1^r & 0 & & \cdots & & \cdots & 0 \\ 0 & \bar{E}_2^r & & \cdots & & \cdots & 0 \\ & & & \vdots & & & \\ 0 & 0 & \cdots & \bar{E}_U^r & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \bar{E}_1^c & \cdots & 0 \\ & & & \vdots & & & \\ 0 & 0 & & \cdots & & & \bar{E}_V^c \end{bmatrix} \tag{40}$$

The constraints of (4)–(6) in the linear programming framework can be also used as the constraints for the quadratic cost function of (39). Only difference is that the elements of s in (39) now take integer values, not the binary ones as in (3), so that the lower bound lb and the upper bound ub represent the range of horizontal and vertical sizes of the grid cells (i.e., the widths of the stripes).

The linear programming framework in (32) – (35) can be reformulated as the constrained optimization with the quadratic cost function

$$s^* = \arg\min_{s} s^T Q^T Q s \tag{41}$$

$$A_{eq} = [1 \ \cdots \ 1] \tag{42}$$

$$b_{eq} = \lceil \alpha \times (M+N) \rceil \tag{43}$$

$$lb = [g_l \ \cdots \ g_l]^T \tag{44}$$

$$ub = [g_{u1} \ \cdots g_{u1} \ g_{u2} \ \cdots \ g_{u2}]^T \tag{45}$$

where the first M elements of (45) take a value g_{u1} and the rest N elements have g_{u2}. Note that, in the linear programming framework of (12) and (13), we set $g_l = 0$ and $g_{u1} = g_{u2} = 1$ for the binary optimization. However, in the above quadratic optimization framework, since s takes a positive integer value to represent the size of the cell in the grid, g_l in (44) and g_{u1}, g_{u2} in (45) are positive integer values such that $g_l < g_{u1}$, $g_l < g_{u2}$, $g_l \ge 0$, $g_{u1} \le \Delta U$, and $g_{u2} \le \Delta V$. Also, the Gaussian weighting factors in (32) can be adopted for each grid-row $G_r(u)$ and grid-column $G_r(v)$ as

$$Q = \begin{bmatrix} \bar{E}_1^r G_r(1) & & & \cdots & 0 \\ & & \vdots & & \\ 0 & \cdots & 0 & \cdots & 0 \\ 0 & \cdots & \bar{E}_1^c G_c(1) & \cdots & 0 \\ & & \vdots & & \\ 0 & \cdots & & & \bar{E}_V^c G_c(V) \end{bmatrix} \tag{46}$$

The quadratic optimization framework in (41) – (46) is formulated to reduce the original image size $M + N$ to $\alpha \times (M + N)$ without the requirement of any specific aspect ratio for the target image, which is similar to the linear programming framework in (32) – (35). Fig. 4 shows the comparative results between the linear and the quadratic optimization frameworks. As you can see in Fig. 4-(e), the resizing effect spreads throughout the entire cells in the same row or column for the alignment, which gives a linear-scaling flavor to the retargeted image. On the other hand, each image-line is independently dropped in the linear programming optimization

in Fig. 4-(c), having a cropping flavor. Also, the linear programming optimization usually converges much faster than the quadratic optimization.

V. FIXED-SIZE AND NEAR-SQUARE IMAGE RESIZING

In this section a case study of the optimal image resizing by the linear programming framework is presented. The problem is originated from the pre-processing task of fitting training and testing images to a fixed-size CNN input. The pre-trained CNN machines that accept only a fixed-size image for both training and testing require the images to be resized as a pre-processing before being fed into the CNN. This resizing process affects the performance of the CNN and it has been shown in [11] that the image classification performance can be improved substantially by the resizing step alone.

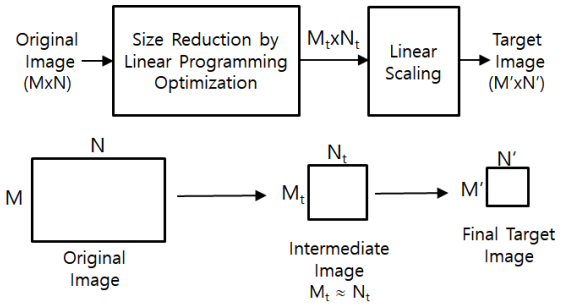

FIGURE 5. Two-stage Image resizing for the fixed-size and square CNN input, (i) First stage: resizing to a near-square image by the line-based linear programming, (ii) Second stage: resizing to the final (square) fixed-size image by linear scaling technique.

The image cropping and linear scaling techniques can be used for the pre-processing task. However, when there exists a big size-difference between the original image and the CNN input, the chance to keep the whole OOI in the resized image by the cropping will be low. Also, for an image with a big difference in aspect ratio between the original image and the CNN input, the linear scaling method may distort the geometric information of the objects in the original image. These problems can be alleviated by properly combining the linear scaling and the cropping. For example, the original image is linearly scaled such that the shorter side is scaled to a fixed one S_c (e.g., $S_c = 256$) and, then, the longer side of the image is scaled accordingly to preserve the aspect ratio. Subsequently, the scaled image is cropped to the dimension of 256×256 and is linearly scaled to the final image size, say 224×224 [10]. The size S_c can be determined by calculating the average size of training images [21]. Also, it can be chosen randomly from a given range, say $S_c \in [256, 512]$. However, these methods of choosing S_c have no consideration on the content as well as the dimension (i.e., size and aspect ratio) of the individual image to be resized. A more desirable approach would take a two-stage approach as shown in Fig. 5, where the first stage is to resize the original image to fit the aspect ratio of the CNN input as much as possible. Here, a constrained optimization can be applied to optimally balance the saliency of the image content and the target aspect ratio of the CNN input as a soft constraint, where the non-salient regions in the original image are the main target for the adjustment of the aspect ratio. As a result of the first-stage, the intermediate image has almost a similar aspect ratio to the CNN input and the subsequent stage of the linear scaling to the final image size will barely distort the shape information of the objects due to the subsequent linear scaling process.

Let us fix the size of the CNN input to 224×224 as VGG Net, then the first stage in Fig. 5 is to resize the original image to be near-square. The line-based linear programming approach introduced in the previous sections is especially useful for resizing a non-square image to a squared one. Note that the line-based optimization can be readily formulated to prune more image lines in the longer side of the image than the shorter one. Therefore, after the original image is resized to a near-square image as a soft requirement by the linear programming optimization, then a simple linear-scaling technique can be applied to the near-square image to have the final square image with the hard (fixed) size requirement for the CNN input (see Fig. 5). Adopting this two-stage strategy, the resizing of the original image with $M \times N$ to the intermediate image with $M_t \times N_t$, where $M_t \approx N_t$, $M_t \leq M$, and $N_t \leq N$, can be done by the linear programming framework. Specifically, an optimal near-square image resizing can be accomplished by using the cost function of (32) with the aspect ratio constraints in (20) and (21) and with the soft constraint of the lower bound for the number of lines to be deleted as follows

$$\begin{aligned} f &= [-E_r(1)G_r(1)\cdots - E_r(M)G_r(M) \\ &\quad - E_c(1)G_c(1)\cdots - E_c(N)G_c(N)] \qquad (47) \\ s &= [s(1)\cdots s(M)\ s(M+1)\cdots s(M+N)] \qquad (48) \\ A &= \begin{bmatrix} \gamma & \cdots & \gamma & -1 & \cdots & -1 \\ -\gamma & \cdots & -\gamma & 1 & \cdots & 1 \\ 1 & \cdots & 1 & 1 & \cdots & 1 \end{bmatrix} \qquad (49) \\ b &= \begin{bmatrix} \beta \times |M-N| \\ \beta \times |M-N| \\ \alpha \times (M+N) \end{bmatrix} \qquad (50) \\ lb &= [0\ \cdots\ 0]^T \qquad (51) \\ ub &= [1\ \cdots\ 1]^T \qquad (52) \end{aligned}$$

Top two rows in (49) and (50) correspond to (20) and (21) and are responsible to meet the requirement of the aspect ratio γ as much as possible, where the tolerable margin toward the exact aspect ratio γ is given by $\beta \times |M-N|$. The parameter β, which is less than 1, controls the amount of tolerable image lines as a margin to the exact γ. The third row in (49) and (50) is to set the lower bound for the number of image lines to be deleted. That is, the total number of remaining lines should be less than $\alpha \times (M+N)$. In other words, α controls the minimal number of image lines to be deleted. Therefore, the above linear programming process optimally selects at least $(1-\alpha) \times (M+N)$ image lines to be deleted and further selects the lines until the reduced image satisfies the aspect ratio γ with

a margin of $\pm\beta \times |M - N|$. Again, the Gaussian weighting factors G_r and G_c in (47) force the optimization to choose the image lines to be deleted far away from the center of the image (i_c, j_c).

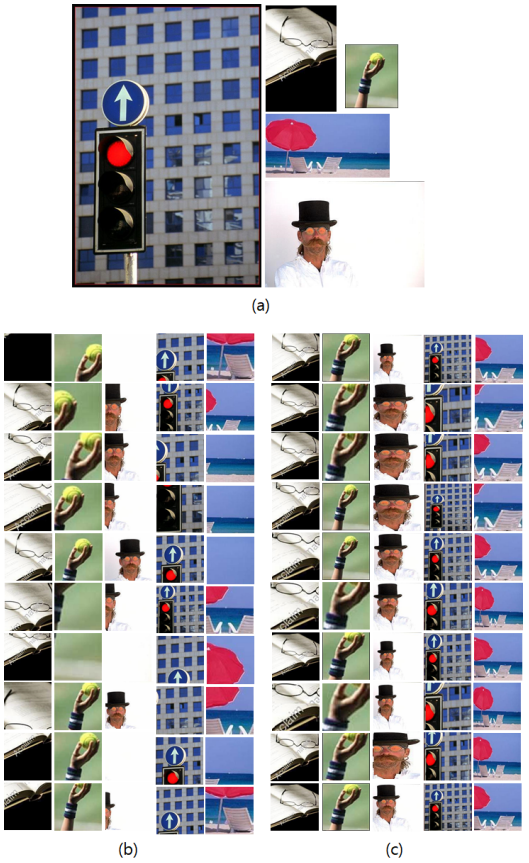

FIGURE 6. 10 consecutive random trials for image resizing to 224 × 224, (a) Original images from Caltech-256 dataset [35], (b) Scale Jittering and Random Cropping (SJRC) method of [11] with $S_c \in [256, 512]$, (c) Near square Line pruning and Linear scaling (NsLpLs) of (47) – (52) for the first stage of Fig. 5 with $\alpha \in [0.7, 1]$ and $\beta \in [0.01, 1]$.

The above linear programming framework can be used for CNN training and testing. When it is used as a pre-processing for CNN training, the parameters α and β in (50) can be chosen randomly to provide the jittering effect. For example, if the parameters are randomly chosen from $\alpha \in [0.7, 1]$ and $\beta \in [0.01, 1]$ for every epoch of CNN training, then the constraints in (50) for the number of image lines to be deleted (i.e., $1-\alpha$) and the tolerable image lines for the given aspect ratio γ vary from 0 to 30% of $(M + N)$ and from 1% to 100% of $|M - N|$, respectively. The randomly chosen pairs of α and β make the relative influence of scaling and the cropping effects to the final images of $M' \times N'$ variable. Note that, for some parameter pairs of α and β, no optimal solution of the above linear programming formulation may be found. In this case, we just set $M_t = M$ and $N_t = N$ for the next linear scaling process. In Fig. 6, 10 consecutive random trials of the two-stage image resizing to 224 × 224 by the linear programming optimization are compared with the scale jittering method of [11] with $S_c \in [256, 512]$. As one can see in the figures the two-stage method with the linear programming framework includes the OOI more reliably in the target 224 × 224 images than the scale jittering method. For CNN testing, three fixed scales such as $S_c \in \{256, 384, 512\}$ are recommended in [11]. Similarly, the parameters for the linear programming optimization can be fixed for CNN testing as $\alpha = 0.8$ and $\beta = 0.2$.

The two image resizing methods in Fig. 6 are applied to the existing CNN architecture for further comparisons. The pre-trained VGG-19 [11] is adopted for the fine-tuning with image dataset of the Caltech-256 [35] and PASCAL VOC 2007 [36]. VGG-19 has 19 layers, where last 3 fully connected layers are used for fine-tuning. Caltech-256 has 30607 images in 257 classes and three random splits of which contains 60 training images per class and the rest for testing are used for experiments. Pascal VOC 2007 [36] involves 9963 images in 20 categories, where 5011 images are for training, and the rest are for testing. No annotations including the object bounding boxes provided by the VOC 2007 have been used for training and testing. No other pre-processing for image augmentation such as random horizontal flipping and RGB jittering have been used so that the training and the testing are solely affected by the image resizing method. In this study, the batch size was set to 64, the number of training epochs was 20, the base learning rate was 0.001, and was decreased by factor of 0.5 after every 5 epochs.

The fine-tuned nets are used for testing. Three fixed scales of $Q = \{256, 384, 512\}$ are applied for the net trained by the scale-jittering method and the final decision is made by aggregating the features in the final fully connected layer [11]. The network trained by the two-stage method of the linear programming framework of (47)–(52) is tested by fixing the parameters as $\alpha = 0.8$ and $\beta = 0.2$. Since multiple objects from multiple classes may exist in the same image for the VOC 2007, it is considered as a correct classification if any one of the objects in the image is classified correctly.

TABLE 1. Classification results of NsLpLs (Near Square Line pruning and Linear scaling) and SJRC (Scaling Jittering and Random Cropping) / MSC (Multi Scaling and Corpping for testing) methods.

DataSet	Image Resizing for VGG-19 Fine-Tuning	Image Resizing for VGG-19 Testing	meanAP
Pascal VOC2007	SJRC S∈[256,512]	MSC Q={256,384,512}	86.80
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**88.17**
Caltech 256	SJRC S∈[256,512]	MSC Q={256,384,512}	80.07 ±0.18
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**81.57** ±0.26

The performance is evaluated by the mean Average Precision (mAP). Table 1 summarizes the testing results. As one can see in Table 1 and Table 2 the Near square Line pruning and Linear scaling (NsLpLs) method of (47)–(52) outperforms the Scale Jittering and Random Cropping (SJRC)

TABLE 2. Comparisons with separated test images of low aspect ratio (lowAR) and high aspect ratio (highAR). Those images with the long-side larger than the short-side more than 1.5 time are grouped at highAR.(∗ denotes the average of the three random splits).

DataSet (Number of Images)	Image Resizing for VGG-19 Fine-Tuning	Image Resizing for VGG-19 Testing	meanAP
Pascal VOC2007-lowAR (3473)	SJRC S∈[256,512]	MSC Q={256,384,512}	87.10
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**88.16**
Pascal VOC2007-highAR (1479)	SJRC S∈[256,512]	MSC Q={256,384,512}	85.29
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**87.81**
Caltech256-lowAR (11,026*)	SJRC S∈[256,512]	MSC Q={256,384,512}	80.20 ±0.27
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**81.26** ±0.28
Caltech256-highAR (4,161*)	SJRC S∈[256,512]	MSC Q={256,384,512}	76.83 ±0.66
	NsLpLs α∈[0.7,1], β∈[0.01,1]	NsLpLs α=0.8, β=0.2	**80.22** ±1.24

for training and the Multi Scaling and Corpping (MSC) for testing method [11]. In particular, the meanAP differences between the NsLpLs and the SJRC/MSC of the images with large aspect ratios (i.e., Pascal VOC2007-highAR and Caltech256-highAR) are much greater than those of low aspect ratios (i.e., Pascal VOC2007-lowAR and Caltech256-lowAR), which clearly shows the effect of adopting the soft condition of the near-square optimization in the NsLpLs.

Note that, since the NsLpLs is based on the iterative linear programming method, it takes more time to finish the image resizing than the SJRC method. Also, the comparative experiments reported in this section are not certainly exhaustive. The experiments should be interpreted as an example of the generalized optimization framework with a soft constraint. The complete algorithm based on the optimization framework for the image resizing as the CNN pre-processing task is left as the future work.

VI. CONCLUSION

As the image resizing problem extends its scope from the display for human vision into the feature preservation for the machine vision, the hard requirements for image parameters such as the size and the aspect ratio can be treated more flexibly as soft conditions. This implies that the determination of the conflicting image parameters becomes more complex and needs a generalized optimization framework to accommodate the various combinations of the diverse requirements. The contributions of this paper can be summarized as follows. First, this paper has formulated the image resizing as a line-based optimization problem via the linear and quadratic programming frameworks. Second, under the optimization framework, some novel combinations of the cost and the constraints are introduced. Specifically, the number of image-lines (i.e., the size of image) can be maximally deleted under a lower bound constraint of the saliency level, which can guarantee a certain level of image saliency. The sparseness and the denseness among the would-be-deleted image lines can be optimally balanced by forcing an alternate line-deleting as a constraint. Also, the soft requirement such as the near-square image size can be obtained by the optimal balance between the image saliency and the squareness of the target image. Finally, as a case study, the near-square image resizing based on the optimization framework is applied to the pre-processing problem for the fixed-size CNN input.

REFERENCES

[1] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," *ACM Trans. Graph.*, vol. 26, no. 3, p. 10, 2007.
[2] M. A. A. Rubinstein, A. Shamir and S. Avidan, "Improved seam carving for video retargeting," *ACM Trans. Graph.*, vol. 27, no. 3, 2008, Art. no. 16.
[3] M. Rubinstein, A. Shamir, and S. Avidan, "Multi-operator media retargeting," *ACM Trans. Graph.*, vol. 28, no. 3, 2009, Art. no. 23.
[4] D. Panozzo, O. Weber, and O. Sorkine, "Robust image retargeting via axis-aligned deformation," *Comput. Graph. Forum*, vol. 31, no. 2pt1, pp. 229–236, May 2012.
[5] J. Lei, M. Wu, C. Zhang, F. Wu, N. Ling, and C. Hou, "Depth-preserving stereo image retargeting based on pixel fusion," *IEEE Trans. Multimedia*, vol. 19, no. 7, pp. 1442–1453, Jul. 2017.
[6] W. Tan, B. Yan, K. Li, and Q. Tian, "Image retargeting for preserving robust local feature: Application to mobile visual search," *IEEE Trans. Multimedia*, vol. 18, pp. 128–137, Jan. 2016.
[7] B. Yan, W. Tan, K. Li, and Q. Tian, "Codebook guided feature-preserving for recognition-oriented image retargeting," *IEEE Trans. Image Process.*, vol. 26., no. 5, pp. 2454–2465, May 2017.
[8] Y. Xia, L. Zhang, R. Hong, L. Nie, Y. Yan, and S. Ling, "Perceptually guided photo retargeting," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 566–578, Mar. 2012.
[9] Y. Zhou, L. Zhang, C. Zhang, P. Li, and X. Li, "Perceptually aware image retargeting for mobile devices," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2301–2313, May 2018.
[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[11] K. Simonyan and A. Zisserman. (2015). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[12] M. B. Lopez, C. R. del-Blanco, and N. Garcia, "Detecting exercise-induced fatigue using thermal imaging and deep learning," in *Proc. 7th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov./Dec. 2017, pp. 1–6.
[13] N. C. Camgoz, S. Hadfield, O. Koller, and R. Bowden, "SubUNets: End-to-end hand shape and continuous sign language recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3075–3084.
[14] A. I. Shahin, Y. Guo, K. M. Amin, and A. A. Sharawi, "White blood cells identification system based on convolutional deep neural learning networks," *Comput. Methods Programs Biomed.*, to be published, doi: 10.1016/j.cmpb.2017.11.015.
[15] T. Chugh, K. Cao, and A. K. Jain, "Fingerprint spoof buster: Use of minutiae-centered patches," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 9, pp. 2190–2202, Sep. 2018.
[16] Z. Zheng, L. Zheng, and Y. Yang, "A discriminatively learned CNN embedding for person reidentification," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 14, no. 1, Jan. 2017, Art. no. 13.
[17] S. Taheri and Ö. Toygar, "Animal classification using facial images with score-level fusion," *IET Comput. Vis.*, vol. 12, no. 5, pp. 679–685, 2018.
[18] W.-T. Sun, T.-H. Chao, Y.-H. Kuo, and W. H. Hsu, "Photo filter recommendation by category-aware aesthetic learning," *IEEE Trans. Multimedia*, vol. 19, no. 8, pp. 1870–1880, Aug. 2017.
[19] J. Tang, L. Jin, Z. Li, and S. Gao, "RGB-D object recognition via incorporating latent data structure and prior knowledge," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 1899–1908, Nov. 2015.

[20] L. Baraldi, C. Grana, and R. Cucchiara, "Recognizing and presenting the storytelling video structure with deep multimodal networks," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 955–968, May 2017.
[21] L. Zheng, Y. Zhao, S. Wang, J. Wang, and Q. Tian. (2016). "Good practice in CNN feature transfer." [Online]. Available: https://arxiv.org/abs/1604.00133
[22] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," in *Proc. IEEE. CVPR*, Jun./Jul. 2004, p. 178.
[23] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. 10th Eur. Conf. Comput. Vis.*, Oct. 2008, pp. 304–317.
[24] D. T. Vo, J. Sole, P. Yin, C. Gomila, and T. Q. Nguyen, "Selective data pruning-based compression using high-order edge-directed interpolation," *IEEE Trans. Image Process.*, vol. 19, no. 2, pp. 399–409, Feb. 2010.
[25] C. S. Won and S.-W. Jung, "Near-reversible efficient image resizing for devices supporting different spatial resolutions," *J. Supercomput.*, vol. 73, no. 7, pp. 3021–3037, 2017.
[26] Y.-S. Wang, C.-L. Tai, O. Sorkine, and T.-Y. Lee, "Optimized scale-and-stretch for image resizing," *ACM Trans. Graph.*, vol. 27, no. 5, 2008, Art. no. 118.
[27] R. Chen, D. Freedman, Z. Karni, C. Gotsman, and L. Liu, "Content-aware image resizing by quadratic programming," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2010, pp. 1–8.
[28] L. Wang, S. Guo, W. Huang, Y. Xiong, and Y. Qiao, "Knowledge guided disambiguation for large-scale scene classification with multi-resolution CNNs," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 2055–2068, Apr. 2017.
[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, Jun. 2014, pp. 580–587.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.
[31] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.
[32] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit. (FG)*, 2017, pp. 650–657.
[33] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast R-CNN for pedestrian detection," *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 985–996, Apr. 2018.
[34] C. S. Won and R. M. Gray, *Stochastic Image Processing*. New York, NY, USA: Springer, 2013.
[35] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep., 2007. [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/index.html
[36] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. (2007). *The PASCAL Visual Object Classes Challenge 2007 (VOC2007), Results*. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html

CHEE SUN WON received the B.S. degree in electronics engineering from Korea University, Seoul, South Korea, in 1982, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Massachusetts, Amherst, MA, USA, in 1986 and 1990, respectively. From 1989 to 1992, he was a Senior Engineer with GoldStar Co., Ltd. (LG Electronics), Seoul. In 1992, he joined Dongguk University, Seoul, where he is currently a Professor with the Division of Electrical and Electronics Engineering. He has been a Visiting Professor at Stanford University, Stanford, CA, USA, and at McMaster University, Hamilton, ON, Canada. His research interests include MRF image modeling, image compressions, content-based image retrieval, and image/video resizing.