

# Recurrent Scale Approximation for Object Detection in CNN

Yu Liu<sup>1,2</sup>, Hongyang Li<sup>2</sup>, Junjie Yan<sup>1</sup>, Fangyin Wei<sup>1</sup>, Xiaogang Wang<sup>2</sup>, Xiaoou Tang<sup>2</sup>

<sup>1</sup>SenseTime Group Limited

<sup>2</sup>Multimedia Laboratory at The Chinese University of Hong Kong

liuyuisanai@gmail.com, {yangli, xgwang}@ee.cuhk.edu.hk,  
{yanjunjie, weifangyin}@sensetime.com, xtang@ie.cuhk.edu.hk

## Abstract

Since convolutional neural network (CNN) lacks an inherent mechanism to handle large scale variations, we always need to compute feature maps multiple times for multi-scale object detection, which has the bottleneck of computational cost in practice. To address this, we devise a recurrent scale approximation (RSA) to compute feature map once only, and only through this map can we approximate the rest maps on other levels. At the core of RSA is the recursive rolling out mechanism: **given an initial map at a particular scale, it generates the prediction at a smaller scale that is half the size of input.** To further increase efficiency and accuracy, we (a): design a scale-forecast network to globally predict potential scales in the image since there is no need to compute maps on all levels of the pyramid. (b): propose a landmark retracing network (LRN) to trace back locations of the regressed landmarks and generate a confidence score for each landmark; LRN can effectively alleviate false positives caused by the accumulated error in RSA. The whole system can be trained end-to-end in a unified CNN framework. Experiments demonstrate that our proposed algorithm is superior against state-of-the-art methods on face detection benchmarks and achieves comparable results for generic proposal generation. The source code of our system is available.<sup>1</sup>

## 1. Introduction

Object detection is one of the most important tasks in computer vision. The convolutional neural network (CNN) based approaches have been widely applied in object detection and recognition with promising performance [10, 13, 15, 18, 22, 23, 27, 34, 36]. To localize objects at arbitrary scales and locations in an image, we need to han-

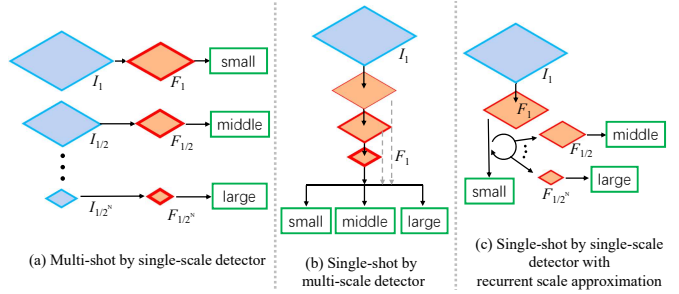


Figure 1. Different detection pipelines. (a) Image pyramid is generated for multi-scale test. The detector only handles a specific range of scales. (b) Image is forwarded once at one scale and the detector generates all results. (c) Our proposed RSA framework. Image is forwarded once only and feature maps for different scales are approximated by a recurrent unit. Blue plates indicate images of different scales and orange plates with red border indicate CNN feature maps at different levels.

dle the variations caused by appearance, location and scale. Most of the appearance variations can now be handled in CNN, benefiting from the invariance property of convolution and pooling operations. The location variations can be naturally solved via sliding windows, which can be efficiently incorporated into CNN in a fully convolutional manner. However, CNN itself does not have an inherent mechanism to handle the scale variations.

The scale problem is often addressed via two ways, namely, multi-shot by single-scale detector and single-shot by multi-scale detector. The first way, as shown in Fig. 1(a), handles objects of different scales independently by resizing the input into different scales and then forwarding the resized images multiple times for detection [2, 16, 28]. Models in such a philosophy probably have the highest recall as long as the sampling of scales is dense enough, but they suffer from high computation cost and more false positives. The second way, as depicted in Fig. 1(b), forwards the image only once and then directly regresses objects at multiple

<sup>1</sup>Our codes and annotations mentioned in Sec.4.1 can be accessed at [github.com/sciencefans/RSA-for-object-detection](https://github.com/sciencefans/RSA-for-object-detection)

scales [21, 26, 27]. Such a scheme takes the scale variation as a black box. Although more parameters and complex structures would improve the performance, the spirit of direct regression still has limitations in real-time applications, for example in face detection, the size of faces can vary from  $20 \times 20$  to  $1920 \times 1080$ .

To handle the scale variation in a CNN-based detection system in terms of both efficiency and accuracy, we are inspired by the fast feature pyramid work proposed by Dollár *et al.* [7], where a detection system using hand-crafted features is designed for pedestrian detection. It is found that image gradients across scales can be predicted based on natural image statistics. They showed that dense feature pyramids can be efficiently constructed on top of coarsely sampled feature pyramids. In this paper, we extend the spirit of fast feature pyramid to CNN and go a few steps further. Our solution to the feature pyramid in CNN descends from the observations of modern CNN-based detectors, including Faster-RCNN [27], R-FCN [4], SSD [21], YOLO [26] and STN [2], where feature maps are first computed and the detection results are decoded from the maps afterwards. However, the computation cost of generating feature maps becomes a bottleneck for methods [2, 28] using multi-scale testing and it seems not to be a neat solution to the scale variation problem.

To this end, our philosophy of designing an elegant detection system is that we calculate the feature pyramid *once* only, and only through that pyramid can we *approximate* the rest feature pyramids at other scales. The intuition is illustrated in Fig. 1(c). In this work, we propose a recurrent scale approximation (RSA, see Fig. 3) unit to achieve the goal aforementioned. The RSA unit is designed to be plugged at some specific depths in a network and to be fed with an initial feature map at the largest scale. The unit convolves the input in a recurrent manner to generate the prediction of the feature map that is half the size of the input. Such a scheme could feed the network with input at one scale only and approximate the rest features at smaller scales through a learnable RSA unit - a balance considering both efficiency and accuracy.

We propose two more schemes to further save the computational budget and improve the detection performance under the RSA framework. The first is a scale-forecast network to globally predict potential scales for a novel image and we compute feature pyramids for just a certain set of scales based on the prediction. There are only a few scales of objects appearing in the image and hence most of the feature pyramids correspond to the background, indicating a redundancy if maps on all levels are computed. The second is a landmark retracing network that retraces the location of the regressed landmarks in the preceding layers and generates a confidence score for each landmark based on the landmark feature set. The final score of identifying a

face within an anchor is thereby revised by the LRN network. Such a design alleviates false positives caused by the accumulated error in the RSA unit.

The pipeline of our proposed algorithm is shown in Fig. 2. The three components can be incorporated into a unified CNN framework and trained end-to-end. Experiments show that our approach is superior to other state-of-the-art methods in face detection and achieves reasonable results for object detection.

To sum up, our contributions in this work are as follows: 1) We prove that deep CNN features for an image can be approximated from different scales using a portable recurrent unit (RSA), which fully leverages efficiency and accuracy. 2) We propose a scale-forecast network to predict valid scales of the input, which further accelerates the detection pipeline. 3) We devise a landmark retracing network to enhance the accuracy in face detection by utilizing landmark information.

## 2. Related work

**Multi-shot by single-scale detector.** A single-scale detector detects the target at a typical scale and cannot handle features at other scales. An image pyramid is thus formulated and each level in the pyramid is fed into the detector. Such a framework appeared in pre-deep-learning era [3, 9] and usually involves hand-crafted features, *e.g.*, HOG [5] or SIFT [24], and some classifier like Adaboost [30], to verify whether the context at each scale contains a target object. Recently, some CNN-based methods [16, 28] also employ such a spirit to predict the objectness and class within a sliding window at each scale. In this way, the detector only handles features in a certain range of scales and the variance is taken over by the image pyramid, which could reduce the fitting difficulty for detector but potentially increase the computational cost.

**Single-shot by multi-scale detector.** A multi-scale detector takes one shot for the image and generates detection results across all scales. RPN [27] and YOLO [26] have fixed size of the input scale, and proposals for all scales are generated in the final layer by using multiple classifiers. However, it is not easy to detect objects in various scales based on the final feature map. Liu *et al.* [21] resolved the problem via a multi-level combination of predictions from feature maps on different scales. And yet it still needs a large model for large receptive field for detection. Other works [17, 20] proposed to merge deep and shallow features in a conv/deconv structure and to merge boxes for objects from different scales. These methods are usually faster than the single-scale detector since it only takes one shot for image, but the large-scale invariance has to be learned by an expensive feature classifier, which is unstable and heavy.

**Face detection.** Recent years have witnessed a performance boost in face detection, which takes advantage of the

use a small net work to detect object scale, then use this scale to generate featuremap(eg if detect  $m=3$ , then resize feature map to  $1./3$  which is from res2b), then use this feature to gen landmark(this may related to face detection) and final result

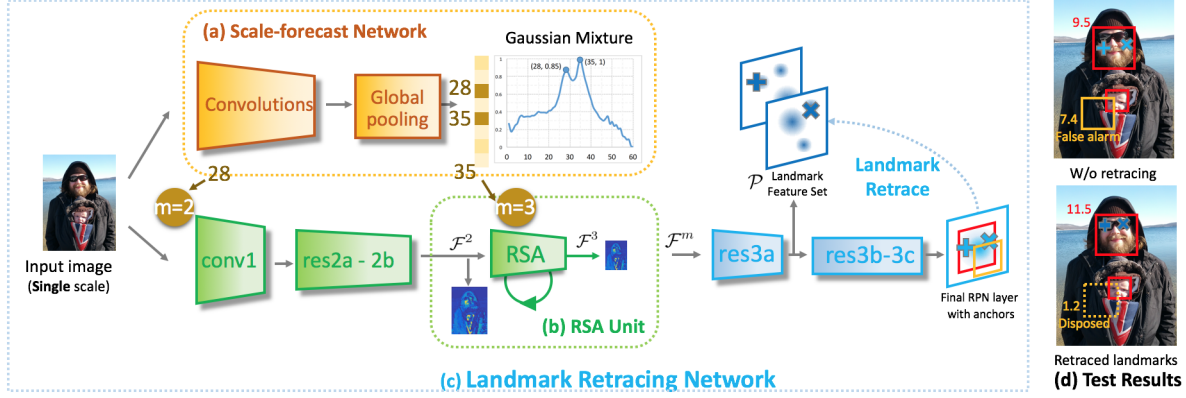


Figure 2. Pipeline of our proposed algorithm. (a) Given an image, we predict potential scales from the scale-forecast network and group the results in six main bins ( $m = 0, \dots, 5$ ). (b) RSA unit. The input is resized based on the smallest scale (corresponding to the largest feature map) and the feature maps at other scales are predicted directly from the unit. (c) Given predicted maps, LRN performs landmark detection in an RPN manner. The landmarks can *trace back* locations via regression to generate individual confidence regarding the existence of the landmark. (d) Due to the retracing mechanism, the final score of detecting a face is revised by the confidence of landmarks, which can effectively dispose of false positives.

development in fully convolutional network [8, 19, 31, 35]. Multi-task RPN is applied [2, 12, 25, 29] to generate face confidence and landmarks together. Both single-scale and multi-scale strategies are introduced in these methods. For example, Chen *et. al* [2] propose a supervised spatial transform layer to utilize landmark information and thus enhance the quality of detector by a large margin.

### 3. Our Algorithm

In this section, we depict each component of our pipeline (Fig. 2) in detail. We first devise a scale-forecast network to predict potential scales of the input; the RSA unit is proposed to learn and predict features at smaller scales based on the output of the scale-forecast network; the image is fed into the landmark retracing network to detect faces of various sizes, using the scale prediction in Section 3.1 and approximation in Section 3.2. The landmark retracing network stated in Section 3.3 can trace back features of regressed landmarks and generate individual confidence of each landmark to revise the final score of detecting a face. At last, we discuss the superiority of our algorithm’s design over other alternatives in Section 3.4.

#### 3.1. Scale-forecast Network

We propose a scale-forecast network (see Fig. 2(a)) to predict the possible scales of faces given an input image of fixed size. The network is a half-channel version of ResNet-18 with a global pooling at the end. The output of this network is a probability vector of  $B$  dimensions, where  $B = 60$  is the predefined number of scales. Let  $\mathcal{B} = \{0, 1, \dots, B\}$  denote the scale set, we define the mapping from a face size  $x$ , in the context of an image being

resized to a higher dimension 2048, to the index  $b$  in  $\mathcal{B}$  as:

$$b = 10(\log_2 x - 5). \quad (1)$$

For example, if the face has size of 64, its corresponding bin index  $b = 10^2$ . Prior to being fed into the network, an image is first resized with the higher dimension equal to 224. During training, the loss of our scale-forecast network is a binary multi-class cross entropy loss:

$$\mathcal{L}^{SF} = -\frac{1}{B} \sum_b p_b \log \hat{p}_b + (1 - p_b) \log(1 - \hat{p}_b), \quad (2)$$

where  $p_b, \hat{p}_b$  are the ground truth label and prediction of the  $b$ -th scale, respectively. Note that the ground truth label for the neighbouring scales  $b_i$  of an occurring scale  $b^*$  ( $p_{b^*} = 1$ ) is not zero and is defined as the Gaussian sampling score:

$$p_{b_i} = \text{Gaussian}(b_i, \mu, \sigma), b_i \in \mathcal{N}(b^*) \quad (3)$$

where  $\mu, \sigma$  are hyperparameters in the Gaussian distribution and  $\mathcal{N}(\cdot)$  denotes the neighbour set. Here we use  $\pm 2$  as the neighbour size and set  $\mu, \sigma$  to  $b^*, 1/\sqrt{2\pi}$ , respectively. Such a practice could alleviate the difficulty of feature learning in the discrete distribution between occurring scales (1) and non-occurring scales (0).

For inference, we use the Gaussian mixture model to determine the local maximum and hence the potential occurring scales. Given observations  $\mathbf{x}$ , the distribution, parameterized by  $\theta$ , can be decomposed into  $K$  mixture components:

$$p(\theta|\mathbf{x}) = \sum_{i=1}^K \phi_i \mathcal{N}(\mu_i, \Sigma_i), \quad (4)$$

<sup>2</sup>Here we assume the minimum and maximum face sizes are 32 and 2048, respectively, if the higher dimension of an image is resized to 2048. In exponential expression, the face size is divided into six main bins from  $2^5 = 32$  to  $2^{11} = 2048$ , the denotation of which will be used later.

where the  $i$ -th component is characterized by Gaussian distributions with weights  $\phi_i$ , means  $\mu_i$  and covariance matrices  $\Sigma_i$ . Here  $K = \{1, \dots, 6\}$  denotes selected scale numbers of six main scales from  $2^5$  to  $2^{11}$  and the scale selection is determined by the threshold  $\phi_i$  of each component. Finally the best fitting model with a specific  $K$  is used.

### 3.2. Recurrent Scale Approximation (RSA) Unit

The recurrent scale approximation (RSA) unit is devised to predict feature maps at smaller scales given a map at the largest scale. Fig. 2 depicts the RSA unit. The network architecture follows a build-up similar to the residual network [13], where we reduce the number of channels in each convolutional layer to half of the original version for time efficiency. The structure details are shown in Section 4.1. Given an input image  $\mathcal{I}$ ,  $\mathcal{I}^m$  denotes the downsampled result of the image with a ratio of  $1/2^m$ , where  $m \in \{0, \dots, M\}$  is the downsample level and  $M = 5$ . Note that  $\mathcal{I}^0$  is the original image. Therefore, there are six scales in total, corresponding to the six main scale ranges defined in the scale-forecast network (see Section 3.1). Given an input image  $\mathcal{I}^m$ , we define the output feature map of layer `res2b` as:

$$f(\mathcal{I}^m) = \mathcal{G}^m, \quad (5)$$

where  $f(\cdot)$  stands for a set of convolutions with a total stride of 8 from the input image to the output map. The set of feature maps  $\mathcal{G}^m$  at different scales serves as the ground truth supervision of the recurrent unit.

The RSA module  $\text{RSA}(\cdot)$  takes as input the feature map of the largest scale  $\mathcal{G}^0$  at first, and repeatedly outputs a map with half the size of the input map:

$$\begin{aligned} h^{(0)} &= \mathcal{F}^0 = \mathcal{G}^0, \\ h^{(m)} &= \text{RSA}(h^{(m-1)}|w) = \mathcal{F}^m. \end{aligned} \quad (6)$$

where  $\mathcal{F}^m$  is the resultant map after being rolled out  $m$  times and  $w$  represents the weights in the RSA unit. The RSA module has four convolutions with a total stride of 2 (1,2,1,1) and their kernel sizes are (1,3,3,1). The loss is therefore the  $l_2$  norm between prediction  $\mathcal{F}^m$  and supervision  $\mathcal{G}^m$  across all scales:

$$\mathcal{L}^{\text{RSA}} = \frac{1}{2M} \sum_{m=1}^M \|\mathcal{F}^m - \mathcal{G}^m\|^2. \quad (7)$$

The gradients in the RSA unit are computed as:

$$\begin{aligned} \frac{\partial \mathcal{L}^{\text{RSA}}}{\partial w_{xy}} &= \sum_m \frac{\partial \mathcal{L}^{\text{RSA}}}{\partial h^{(m)}} \cdot \frac{\partial h^{(m)}}{\partial w_{xy}}, \\ &= \frac{1}{M} \sum_m (\mathcal{F}^m - \mathcal{G}^m) \cdot \mathcal{F}_{xy}^{m-1}, \end{aligned} \quad (8)$$

where  $x$  and  $y$  are spatial indices in the feature map<sup>3</sup>.

The essence behind our RSA unit is to derive a mapping  $\text{RSA}(\cdot) \mapsto f(\cdot)$  to constantly predict smaller-scale features based on the current map instead of feeding the network with inputs of different scales for multiple times. In an informal mathematical expression, we have:

$$\lim_{0 \rightarrow m} \text{RSA}(h^{(m-1)}) = f(\mathcal{I}^m) = \mathcal{G}^m,$$

to indicate the functionality of RSA: an approximation to  $f(\cdot)$  from the input at the largest scale 0 to its desired level  $m$ . The computation cost of generating feature map  $\mathcal{F}^m$  using RSA is much lower than that of resizing the image and feeding into the network (*i.e.*,  $f(\mathcal{I}^m)$  through `conv1` to `res2b`; see quantitative results in Section 4.4).

During inference, we first obtain the possible scales of the input from the scale-forecast network. The image is then resized accordingly so that the smallest scale (corresponding to the largest feature map) is resized to the range of [64, 128]. The feature maps at other scales are thereby predicted by the output of RSA unit via Eqn (6). Fig. 3 depicts a rolled-out version of RSA to predict feature maps of smaller scales compared with the ground truth. We can observe from both the error rate and predicted feature maps in each level that RSA is capable of approximating the feature maps at smaller scales.

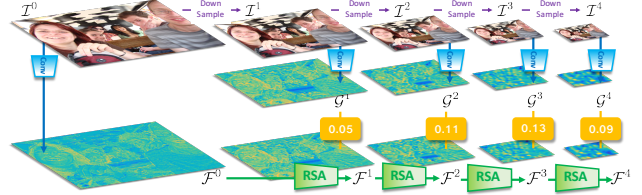


Figure 3. RSA by rolling out the learned feature map at smaller scales. The number in the orange box indicates the average mean squared error between ground truth and RSA’s prediction.

### 3.3. Landmark Retracing Network

In the task of face detection, as illustrated in Fig. 2, the landmark retracing network (LRN) is designed to adjust the confidence of identifying a face and to dispose of false positives by learning individual confidence of each regressed landmark. Instead of directly using the ground truth location of landmarks, we formulate such a feature learning of landmarks based on the regression *output* of landmarks in the final RPN layer.

Specifically, given the feature map  $\mathcal{F}$  at a specific scale from RSA ( $m$  is dropped for brevity), we first feed it into the `res3a` layer. There are two branches at the output:

<sup>3</sup>For brevity of discussion, we ignore the spatial weight sharing of convolution here. Note that the weight update in  $w_{xy}$  also includes the loss from the landmark retracing network.



one is the landmark feature set  $\mathcal{P}$  to predict the individual score of each landmark in a spatial context. The number of channels in the set equals to the number of landmarks. Another branch is continuing the standard RPN [27] pipeline (res3b-3c) which generates a set of anchors in the final RPN layer. Let  $\mathbf{p}_i = [p_{i0}, p_{i1}, \dots, p_{ik}, \dots]$  denote the classification probability in the final RPN layer, where  $k$  is the class index and  $i$  is the spatial location index on the map;  $t_{ij}$  denotes the regression target (offset defined in [10]) of the  $j$ -th landmark in the  $i$ -th anchor, where  $j = \{1, \dots, 5\}$  is the landmark index. Note that in face detection task, we only have one anchor so that  $\mathbf{p}_i$  contains one element. In the traditional detection-to-landmark formulation, the following loss, which consists of two heads (*i.e.*, classification and regression), is optimized:

$$\sum_i -\log p_{ik^*} + \delta(k^*) \mathcal{S}(t_i - t_i^*),$$

where  $\delta(\cdot)$  is the indicator function;  $k^*$  denotes the correct label of anchor  $i$  and we have only two classes here (0 for background, 1 for positive);  $t_i^*$  is the ground truth regression target and  $\mathcal{S}(\cdot)$  is the smoothing  $l_1$  loss defined in [10].

However, as illustrated in Fig. 2(c), using the confidence of anchor  $p_{ik^*}$  alone results in false positives in some cases, which inspires us to take advantage of the landmark features based on the regression output. The revised classification output,  $p_{ik^*}^{trace}(t_{ij})$ , now considers both the feature in the final RPN layer as well as those in the landmark feature set:

$$p_{ik^*}^{trace}(t_{ij}) = \begin{cases} p_{i0}, & k^* = 0, \\ \max\_pool(p_{i1}, p_{ij}^{land}), & k^* = 1, \end{cases} \quad (9)$$

where  $p_{ij}^{land}$  is the classification output of point  $j$  from the landmark feature set  $\mathcal{P}$  and it is determined by the regression output:

$$p_{ij}^{land} = \mathcal{P}(r(t_{ij})), \quad (10)$$

where  $r(\cdot)$  stands for a mapping from the regression target to the spatial location on map  $\mathcal{P}$ . To this end, we have the revised loss for our landmark retracing network:

$$\mathcal{L}^{LRN} = \sum_i \left[ -\log p_{ik^*}^{trace}(t_{ij}) + \delta(k^*) \sum_j \mathcal{S}(t_{ij} - t_{ij}^*) \right]. \quad (11)$$

Apart from the detection-to-landmark design as previous work did, our retracing network also fully leverages the feature set of landmarks to help rectify the confidence of identifying a face. This is achieved by utilizing the regression output  $t_{ij}$  to find the individual score of each landmark on the preceding feature map  $\mathcal{P}$ . Such a scheme is in a landmark-to-detection spirit.

Note that the landmark retracing network is trained end-to-end with the RSA unit stated previously. The anchor associated with each location  $i$  is a square box of fixed size

$64\sqrt{2}$ . The landmark retracing operation is performed only when the anchor is a positive sample. The base landmark location with respect to the anchor is determined by the average location of all faces in the training set. During test, LRN is fed with feature maps at various scales and it treats each scale individually. The final detection result is generated after performing NMS among results from multi-scales.

### 3.4. Discussion

**Comparison to RPN.** The region proposal network [27] takes a set of predefined anchors of different sizes as input and conducts a similar detection pipeline. Anchors in RPN vary in size to meet the multi-scale training constraint. During one iteration of update, it has to feed the whole image of different sizes (scales) from the start to the very end of the network. In our framework, we resize the image *once* to make sure at least one face falls into the size of  $[64, 128]$ , thus enforcing the network to be trained within a certain range of scales. In this way, we are able to use only one anchor of fixed size. The multi-scale spirit is embedded in an RSA unit, which directly predicts the feature maps at smaller scales. Such a scheme saves parameters significantly and could be considered as a ‘semi’ multi-scale training and ‘fully’ multi-scale test.

**Prediction-supervised or GT-supervised in landmark feature sets.** Another comment on our framework is the supervision knowledge used in training the landmark features  $\mathcal{P}$ . The features are learned using the prediction output of regression targets  $t_{ij}$  instead of the ground truth targets  $t_{ij}^*$ . In our preliminary experiments, we find that if  $p_i^{land} \sim t_i^*$ , the activation in the landmark features would be heavily suppressed due to the misleading regression output by  $t_{ij}$ ; however, if we relax the learning restriction and accept activations within a certain range of misleading locations, *i.e.*,  $p_i^{land} \sim t_i$ , the performance can be boosted further. Using the prediction of regression as supervision in the landmark feature learning makes sense since: (a) we care about the activation (classification probability) rather than the accurate location of each landmark; (b)  $t_i$  and  $p_i^{land}$  share similar learning workflow and thus the location of  $t_i$  could better match the activation  $p_i^{land}$  in  $\mathcal{P}$ .

## 4. Experiments

In this section we first conduct the ablation study to verify the effectiveness of each component in our method and compare exhaustively with the baseline RPN [27]; then we compare our algorithm with state-of-the-art methods in face detection and object detection on four popular benchmarks.

### 4.1. Setup and Implementation Details

Annotated Faces in the Wild (AFW) [37] contains 205 images for evaluating face detectors’ performance. However, some faces are missing in the annotations and could

trigger the issue of false positives, we relabel those missing faces and report the performance difference in both cases. Face Detection Data Set and Benchmark (Fddb) [14] has 5,171 annotated faces in 2,845 images. It is larger and more challenging than AFW. Multi-Attribute Labelled Faces (MALF) [32] includes 5,250 images with 11,931 annotated faces collected from the Internet. The annotation set is cleaner than that of AFW and it is the largest benchmark for face detection.

Our training set has 184K images, including 171K images collected from the Internet and 12.9K images from the training split of Wider Face Dataset [33]. All faces are labelled with bounding boxes and five landmarks. The structure of our model is a shallow version of the ResNet [13] where the first seven ResNet blocks are used, *i.e.*, from `conv1` to `res3c`. We use this model in scale-forecast network and LRN. All numbers of channels are set to half of the original ResNet model, for the consideration of time efficiency. We first train the scale-forecast network and then use the output of predicted scales to launch the RSA unit and LRN. Note that the whole system (RSA+LRN) is trained end-to-end and the model is trained from scratch without resorting to a pretrained model since the number of channels is halved. The ratio of the positive and the negative is 1 : 1 in all experiments. The batch size is 4; base learning rate is set to 0.001 with a decrease of 6% every 10,000 iterations. The maximum training iteration is 1,000,000. We use stochastic gradient descent as the optimizer.

## 4.2. Performance of Scale-forecast Network

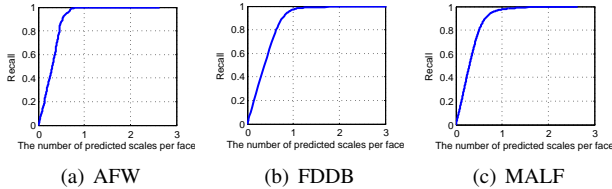


Figure 4. Recall v.s. the number of predicted scales per face on three benchmarks. Our scale-forecast network recalls almost all scales when the number of predicted scale per face is 1.

The scale-forecast network is of vital importance to the computational cost and accuracy in the networks afterwards. Fig. 4 reports the overall recall with different numbers of predicted scales on three benchmarks. Since the number of faces and the number of potential scales in the image vary across datasets, we use *the number of predicted scales per face* ( $x$ , total predicted scales over total number of faces) and a global *recall* ( $y$ , correct predicted scales over all ground truth scales) as the evaluation metric. We can observe from the results that our trained scale network recalls almost 99% at  $x = 1$ , indicating that on average we only need to generate less than two predictions per image and that we can retrieve all face scales. Based on this prior

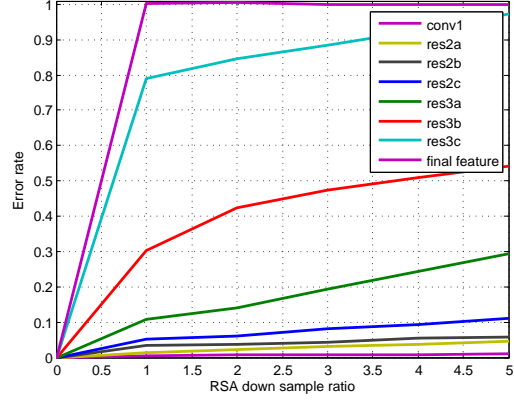


Figure 5. Investigation on the source layer to branch out the RSA unit. For each case, we report the error rate v.s. the level of down-sampling ratio in the unit. We can conclude that the deeper the RSA is branched out, the worse the feature approximation at smaller scales will be.

knowledge, during inference, we set the threshold for predicting potential scales of the input so that it has approximately two predictions.

## 4.3. Ablative Evaluation on RSA Unit

Fig. 5 investigates the effect of appending the RSA unit to different layers. For each case, the error rate between the ground truth and corresponding prediction is computed. We define the error rate (ER) on level  $m$  as:

$$ER^m = \frac{1}{N} \sum_i^N ((\mathcal{F}^{(m)} - \mathcal{G}^{(m)}) ./ \mathcal{G}^{(m)})^2, \quad (12)$$

where ‘./’ implies an element-wise division between maps;  $N$  is the total number of samples. We use a separate validation set to conduct this experiment. The image is first resized to higher dimension being 2048 and the RSA unit predicts six scales defined in Section 3.1 (1024, 512, 256, 128 and 64). Ground truth maps are generated accordingly as we iteratively resize the image (see Fig. 3). There are two remarks regarding the result:

First, *feature depth matters*. Theoretically RSA can handle all scales of features in a deep CNN model and therefore can be branched out at any depth of the network. However, results from the figure indicate that as we plug RSA at deeper layers, its performance decays. Since features at deeper layers are more sparse and abstract, they barely contain information for RSA to approximate the features at smaller scales. For example, in case *final feature* which means RSA is plugged at the final convolution layer after `res3c`, the error rate is almost 100%, indicating RSA’s incapability of handling the insufficient information in this layer. The error rate decreases in shallower cases.

However, the computation cost of RSA at shallow layers is much higher than that at deeper layers, since the stride is

Table 1. The proposed algorithm is more computationally efficient and accurate by design than baseline RPN. Theoretical operations of each component are provided, denoted as ‘Opts. (VGA input)’ below. The minimum operation in each component means only the scale-forecast network is used where no face appears in the image; and the maximum operation indicates the amount when faces appear at all scales. The actual runtime comparison between ours and baseline RPN is reported in Table 2.

Component	Scale-forecast	RSA	LRN	Total Pipeline	Baseline RPN	
Structure	tiny ResNet-18	4-layer FCN	tiny ResNet-18	-	single anchor	multi anchors
Opts. (VGA input)	95.67M	0 to 182.42M	0 to 1.3G	<b>95.67M to 1.5G</b>	1.72G	1.31G
AP@AFW	-	-	-	<b>99.96%</b>	99.90%	98.29%
Recall@FDDb1%fp	-	-	-	<b>91.92%</b>	90.61%	86.89%
Recall@MALF1%fp	-	-	-	<b>90.09%</b>	88.81%	84.65%

smaller and the input map of RSA is thus larger. The path during one-time forward from image to the input map right before RSA is shorter; and the rolling out time increases accordingly. Therefore, the trade-off is that we want to plug RSA at shallow layers to ensure a low error rate and at the same time, to save the computational cost. In practice we choose case `res2b` to be the location where RSA is branched out. Most of the computation happens before layer `res2b` and it has an acceptable error rate of 3.44%. We use this setting throughout the following experiments.

Second, *butterfly effect exists*. For a particular case, as the times of the recurrent operation increase, the error rate goes up due to the cumulative effect of rolling out the predictions. For example, in case `res2b`, the error rate is 3.44% at level  $m = 1$  and drops to 5.9% after rolling out five times. Such an increase is within the tolerance of the system and still suffices the task of face detection.

#### 4.4. Our Algorithm vs. Baseline RPN

We compare our model (denoted as RSA+LRN), a combination of the RSA unit and a landmark retracing network, with the region proposal network (RPN) [27]. In the first setting, we use the original RPN with multiple anchors (denoted as  $RPN_m$ ) to detect faces of various scales. In the second setting, we modify the number of anchors to one (denoted as  $RPN_s$ ); the anchor can only detect faces in the range from 64 to 128 pixels. To capture all faces, it needs to take multiple shots in an image pyramid spirit. The network structure of both baseline RPN and our LRN descend from ResNet-18 [13]. Anchor sizes in the first setting  $RPN_m$  are  $32\sqrt{2}, 64\sqrt{2}, \dots, 1024\sqrt{2}$  and they are responsible for detecting faces in the range of  $[32, 64), [64, 128), \dots, [1024, 2048]$ , respectively. In the second setting  $RPN_s$ , we first resize the image length to 64, 256,  $\dots$ , 2048, then test each scale individually and merge all results through NMS [1].

Table 1 shows the theoretical computation cost and test performance of our algorithm compared with baseline RPN. We can observe that  $RPN_s$  needs six shots for the same image during inference and thus the computation cost is much larger than ours or  $RPN_m$ ; Moreover,  $RPN_m$  performs worse than the rest two for two reasons: First, the

Table 2. Test runtime (ms per image) of RSA compared with RPN on three benchmarks. We conduct experiments of each case five times and report the average result to avoid system disturbance.

Speed	LRN+RSA	LRN	$RPN_s$	$RPN_m$
AFW	<b>13.95</b>	28.84	26.85	18.92
FDDb	<b>11.24</b>	27.10	25.01	18.34
MALF	<b>16.38</b>	29.73	27.37	19.37
Average	<b>14.50</b>	28.78	26.52	18.99

receptive field is less than 500 and therefore it cannot see the context of faces larger than 500 pixels; second, it is hard for the network (its model capacity much less than the original ResNet [13]) to learn the features of faces in a wide scale range from 32 to 2048.

Table 2 depicts the runtime comparison during test. The third column LRN means without using the RSA unit. Our method runs fast enough compared with its counterparts for two reasons. First, there are often one or two valid scales in the image, and the scale-forecast network can automatically select some particular scales, and ignore all the other invalid ones in the multi-scale test stage; second, the input of LRN descends from the output of RSA to predict feature maps at smaller scales; it is not necessary to compute feature maps of multiple scales in a multi-shot manner as  $RPN_m$  does.

#### 4.5. Face Detection

Fig. 7 shows the comparison against other approaches on three benchmarks. On AFW, our algorithm achieves an AP of 99.17% using the original annotation and an AP of 99.96% using the revised annotation 7(c). On FDDb, RSA+LRN recalls 93.0% faces with 50 false positives 7(a). On MALF, our method recalls 82.4% faces with zero false positive 7(d). It should be noticed that the shape and scale definition of bounding box on each benchmark varies. For instance, the annotation on FDDb is ellipse while others are rectangle. To address this, we learn a transformer to fit each annotation from the landmarks. This strategy significantly enhances performance in the continuous setting on FDDb.

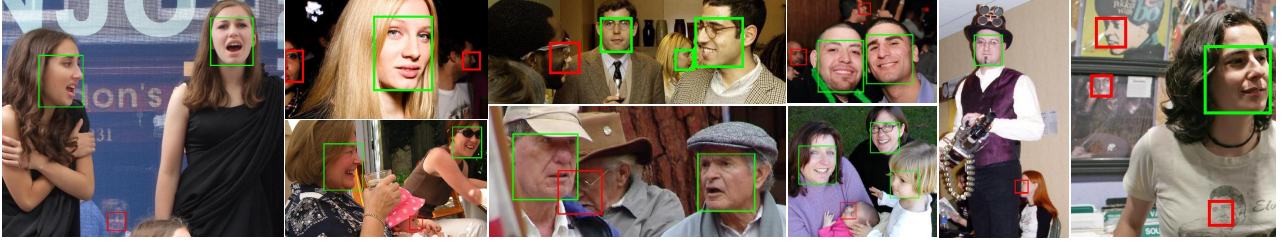


Figure 6. Our proposed model can detect faces at various scales, including the green annotations provided in AFW [37] as well as faces marked in red that are of small sizes and not labeled in the dataset.

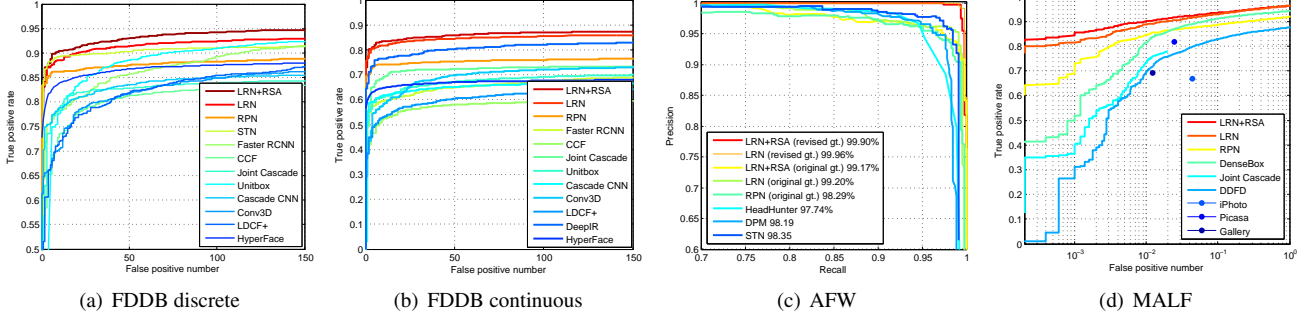


Figure 7. Comparison to state-of-the-art approaches on face detection benchmarks. The proposed algorithm (Scale-forecast network with RSA+LRN, tagged by LRN+RSA) outperforms other methods by a large margin. ‘revised gt.’ and ‘original gt.’ in AFW stand for fully annotated faces by us and partially labeled annotations provided by the dataset, respectively.

Table 3. Recall (%) vs. the number of proposals and Speed (ms per image) breakdown on ILSVRC DET val2.

Recall	100	300	1000	2000	Speed
Original RPN	88.7	93.5	97.3	97.7	158 ms
Single-scale RPN	89.6	94.4	97.2	98.0	249 ms
RSA+RPN	89.1	94.4	97.2	98.0	124 ms

#### 4.6. RSA on Generic Object Proposal

We now verify that the scale approximation learning by RSA unit also generalizes comparably well on the generic region proposal task. Region proposal detection is a basic stage for generic object detection task and is more difficult than face detection. ILSVRC DET [6] is a challenging dataset for generic object detection. It contains more than 300K images for training and 20K images for validation. We use a subset (around 170k images) of the original training set for training, where each category has at most 1000 samples; for test we use the val2 split [11] with 9917 images. We choose the single anchor RPN with ResNet-101 as the baseline. RSA unit is set after res3b3. The anchors are of size  $128\sqrt{2}$  squared,  $128 \times 256$  and  $256 \times 128$ . During training, we randomly select an object and resize the image so that the object is rescaled to  $[128, 256]$ . Scale-forecast network is also employed to predict the higher dimension of objects in the image.

Recalls with different number of proposals are shown in Table 3. The original RPN setting has 18 anchors with 3 as-

pect ratios and 6 scales. Without loss of recall, RPN+RSA reduces around 61.05% computation cost compared with the single-scale RPN, when the number of boxes is over 100. RPN+RSA is also more efficient and recalls more objects than original RPN. Our model and the single-anchor RPN both perform better than the original RPN. This observation is in accordance with the conclusion in face detection. Overall, our scheme of using RSA plus LRN competes comparably with the standard RPN method in terms of computation efficiency and accuracy.

## 5. Conclusion

In this paper, we prove that deep CNN features of an image can be approximated from a large scale to smaller scales by the proposed RSA unit, which significantly accelerates face detection while achieving comparable results in object detection. In order to make the detector faster and more accurate, we devise a scale-forecast network to predict the potential object scales. We further design a landmark retracing network to fuse global and local scale information to enhance the predictor. Experimental results show that our algorithm significantly outperforms state-of-the-art methods. Future work includes exploring RSA on generic object detection task. Representation approximation between video frames is also an interesting research avenue.



## References

- [1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Trans on PAMI*, 2012.
- [2] D. Chen, G. Hua, F. Wen, and J. Sun. Supervised transformer network for efficient face detection. In *ECCV*, pages 122–138. Springer, 2016.
- [3] D. Chen, S. Ren, Y. Wei, X. Cao, and J. Sun. Joint cascade face detection and alignment. In *ECCV*, pages 109–122. Springer, 2014.
- [4] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 379–387. Curran Associates, Inc., 2016.
- [5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision & Pattern Recognition*, pages 886–893, 2005.
- [6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.
- [7] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014.
- [8] S. S. Farfade, M. J. Saberian, and L.-J. Li. Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650. ACM, 2015.
- [9] P. F. Felzenszwalb, R. B. Girshick, and D. Mcallester. Cascade object detection with deformable part models. In *CVPR*. IEEE, 2010.
- [10] R. Girshick. Fast R-CNN. In *ICCV*, 2015.
- [11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.
- [12] Z. Hao, Y. Liu, H. Qin, and J. Yan. Scale-aware face detection. In *CVPR 2017*, 2017.
- [13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [14] V. Jain and E. Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst, 2010.
- [15] B. Leng, Y. Liu, K. Yu, X. Zhang, and Z. Xiong. 3d object understanding with 3d convolutional neural networks. *Information Sciences*, 366:188–201, 2016.
- [16] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua. A convolutional neural network cascade for face detection. In *CVPR*, pages 5325–5334, 2015.
- [17] H. Li, Y. Liu, W. Ouyang, and X. Wang. Zoom out-and-in network with recursive training for object proposal. *arXiv preprint arXiv:1702.05711*, 2017.
- [18] H. Li, Y. Liu, X. Zhang, Z. An, J. Wang, Y. Chen, and J. Tong. Do we really need more training data for object localization. In *IEEE International Conference on Image Processing*, 2017.
- [19] Y. Li, B. Sun, T. Wu, and Y. Wang. Face detection with end-to-end integration of a convnet and a 3d model. In *ECCV*, pages 420–436. Springer, 2016.
- [20] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. *arXiv preprint arXiv:1612.03144*, 2016.
- [21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016.
- [22] Y. Liu, H. Li, and X. Wang. Learning deep features via congenerous cosine loss for person recognition. *arXiv preprint arXiv:1702.06890*, 2017.
- [23] Y. Liu, J. Yan, and W. Ouyang. Quality aware network for set to set recognition. In *CVPR*, 2017.
- [24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [25] R. Ranjan, V. M. Patel, and R. Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *arXiv preprint arXiv:1603.01249*, 2016.
- [26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the CVPR*, pages 779–788, 2016.
- [27] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*, 2015.
- [28] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.
- [29] X. Sun, P. Wu, and S. C. Hoi. Face detection using deep learning: An improved faster rcnn approach. *arXiv preprint arXiv:1701.08289*, 2017.
- [30] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.
- [31] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features. In *Proceedings of the IEEE international conference on computer vision*, pages 82–90, 2015.
- [32] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Fine-grained evaluation on face detection in the wild. In *Automatic Face and Gesture Recognition (FG), 11th IEEE International Conference on*. IEEE, 2015.
- [33] S. Yang, P. Luo, C. C. Loy, and X. Tang. Wider face: A face detection benchmark. In *CVPR*, 2016.
- [34] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *ECCV*, pages 36–42. Springer International Publishing, 2016.
- [35] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang. Unitbox: An advanced object detection network. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 516–520. ACM, 2016.
- [36] X. Zeng, W. Ouyang, J. Yan, H. Li, T. Xiao, K. Wang, Y. Liu, Y. Zhou, B. Yang, Z. Wang, et al. Crafting gbd-net for object detection. *arXiv preprint arXiv:1610.02579*, 2016.
- [37] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, pages 2879–2886, 2012.