Feature Pyramid Networks for Object Detection

Hongzhi Liu

Jun 16, 2018

Abstract

Recognizing objects at vastly different scales is a fundamental challenge in computer vision. Feature pyramids are a basic component in recognition systems for detecting objects at different scales. But recent deep learning object detectors have avoided pyramid representations, in part because they are compute and memory intensive. Today, I read a thesis written by Tsung-Yi Lin, who is from Facebook AI Research. His team exploit the inherent multi-scale, pyramidal hierarchy of deep convolutional networks, called a Feature Pyramid Network (FPN), to construct feature pyramids with marginal extra cost. A top-down architecture with lateral connections is developed for building high-level semantic feature maps at all scales. Using FPN in a basic Faster R-CNN system, our method achieves state-of-the-art single-model results on the COCO detection benchmark without bells and whistles, surpassing all existing single-model entries.

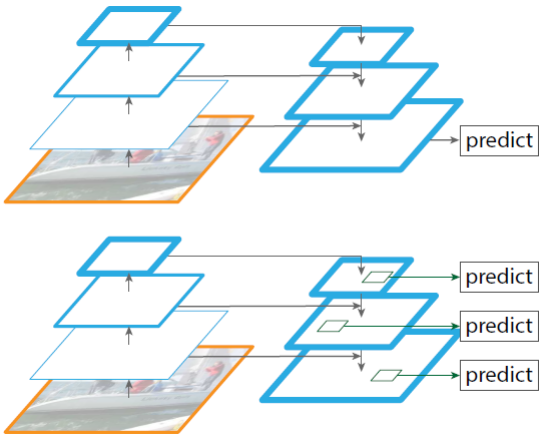

Figure 1. Top: a top-down architecture with skip connections, where predictions are made on the finest level. Bottom: their model that has a similar structure but leverages it as a feature pyramid, with predictions made independently at all levels.

1. Overview of Feature Pyramid Network

HOG and SIFT pyramids have been used in numerous works for image classification, object detection, human pose estimation and more. Before HOG and SIFT, early work on face detection with ConvNets [12] computed shallow networks over image pyramids to detect faces across scales. With the development of modern deep ConvNets [6], object detectors like Over-Feat [11] and R-CNN [3] showed dramatic improvements in accuracy. Because they offer a good trade-off between accuracy and speed. Multi-scale detection, however, still performs better, especially for small objects. There are recent methods exploiting lateral or skip connections that associate low-level feature maps across resolutions and semantic levels. Although these methods adopt architectures with pyramidal shapes, they are unlike featurized image pyramids [1] where predictions are made independently at all levels, see Fig. 1. In fact, for the pyramidal architecture in Fig. 1 (top), image pyramids are still needed to recognize objects across multiple scales [9].

The goal of this paper is to naturally leverage the pyramidal shape of a ConvNet's feature hierarchy while creating a feature pyramid that has strong semantics at all scales. To achieve this goal, Lin relies on an architecture that combines low-resolution, semantically strong features with high-resolution, semantically weak features via a top-down pathway and lateral connections. Lin evaluates their method, called a Feature Pyramid Network (FPN), in various systems for detection and segmentation [7]. They report a state-of-the-art single-model result on the challenging COCO detection benchmark [8] simply based on FPN and a basic Faster R-CNN detector surpassing all existing heavily-engineered single-model entries of competition winners. Furthermore, their method is also easily extended to mask proposals and improves both instance segmentation AR and speed over state-of-the-art methods that heavily depend on image pyramids.

2. Architecture of Feature Pyramid Networks

The goal of team is to leverage a ConvNet's pyramidal feature hierarchy, which has semantics from low to high

levels and build a feature pyramid with high-level semantics throughout. The resulting Feature Pyramid Network is general-purpose and in this paper they focus on sliding window proposers [10] and region-based detectors [2]. Their method takes a single-scale image of an arbitrary size as input, and outputs proportionally sized feature maps at multiple levels, in a fully convolutional fashion. This process is independent of the backbone convolutional architectures and in this paper they present results using ResNets [5]. The construction of the pyramid involves a bottom-up pathway, a top-down pathway, and lateral connections, as introduced in the following.

Bottom-up Pathway. The bottom-up pathway is the feed-forward computation of the backbone ConvNet, which computes a feature hierarchy consisting of feature maps at several scales with a scaling step of 2. There are often many layers producing output maps of the same size and team say these layers are in the same network stage. For their feature pyramid, they define one pyramid level for each stage. They choose the output of the last layer of each stage as their reference set of feature maps, which they will enrich to create our pyramid. This choice is natural since the deepest layer of each stage should have the strongest features.

Top-down pathway and lateral connections. The top-down pathway hallucinates higher resolution features by upsampling spatially coarser, but semantically stronger, feature maps from higher pyramid levels. These features are then enhanced with features from the bottom-up pathway via lateral connections. Fig. 2 shows the building block that constructs our topdown feature maps. With a coarser-resolution feature map, they upsample the spatial resolution by a factor of 2. The upsampled map is then merged with the corresponding bottom-up map by element-wise addition. This process is iterated until the finest resolution map is generated. To start the iteration, we simply attach a 1×1 convolutional layer on C_5 to produce the coarsest resolution map. Finally, the team append a 3×3 convolution on each merged map to generate the final feature map, which is to reduce the aliasing effect of upsampling. This final set of feature maps is called P_2, P_3, P_4, P_5, corresponding to C_2, C_3, C_4, C_5 that are respectively of the same spatial sizes.

Because all levels of the pyramid use shared classifiers as in a traditional featurized image pyramid, the team fix the feature dimension (numbers of channels, denoted as d) in all the feature maps. They set $d = 256$ in this paper and thus all extra convolutional layers have 256-channel outputs. There are no non-linearities in these extra layers, which they have empirically found to have minor impacts.

3. Feature Pyramids Building

The method of team is a generic solution for building feature pyramids inside deep ConvNets. To demonstrate the simplicity and effectiveness of their method, they make minimal modifications to the original systems of [10, 2] when adapting them to their feature pyramid.

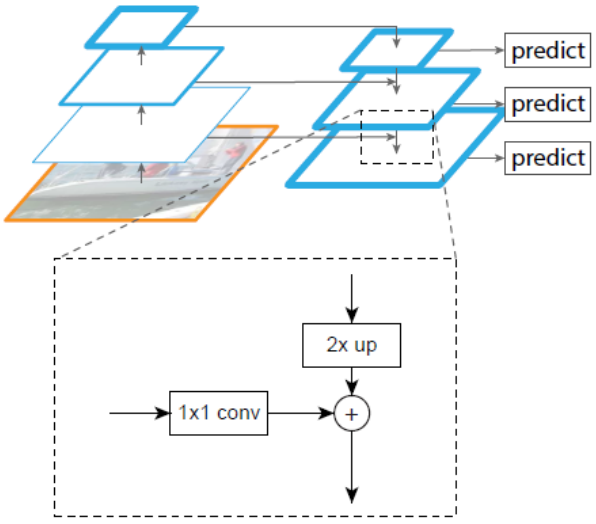

Figure 2. A building block illustrating the lateral connection and the top-down pathway, merged by addition.

3.1. Feature Pyramid Networks for RPN

Lin adapts RPN by replacing the single-scale feature map with our FPN. He attaches a head of the same design (3×3 conv and two sibling 1×1 convs) to each level on their feature pyramid. Because the head slides densely over all locations in all pyramid levels, it is not necessary to have multi-scale anchors on a specific level. Instead, they assign anchors of a single scale to each level. Formally, they define the anchors to have areas of $32^2, 64^2, 128^2, 256^2, 512^2$ pixels on P_2, P_3, P_4, P_5, P_6 respectively. As in [10] they also use anchors of multiple aspect ratios $1:2, 1:1, 2:1$ at each level. So in total there are 15 anchors over the pyramid.

With the above adaptations, RPN can be naturally trained and tested with their FPN, in the same fashion as in [10]. Lin elaborates on the implementation details in the experiments.

3.2. Feature Pyramid Networks for Fast RCNN

Fast R-CNN [2] is a region-based object detector in which Region-of-Interest (RoI) pooling is used to extract features. Fast R-CNN is most commonly performed on a single-scale feature map. To use it with their FPN, they need to assign RoIs of different scales to the pyramid levels. They view their feature pyramid as if it were produced from an image pyramid. Thus they can adapt the assignment strategy of region-based detectors [2, 4] in the case when running on image pyramids. Formally, they assign an RoI of width w and height h to the level P_k of our feature pyramid as Equation 1:

Table 1. Object detection results using Faster R-CNN [10] evaluated on the COCO minival set. The backbone network for RPN are consistent with Fast R-CNN. Models are trained on the trainval35k set and use ResNet-50. Provided by authors of [5].

Faster R-CNN	proposals	feature	head	lateral?	top-down?	AP@0.5	AP	AP_s	AP_m	AP_l
baseline from He et al. [5]	RPN, C_4	C_4	conv5			47.3	26.3	-	-	-
(a) baseline on conv4	RPN, C_4	C_4	conv5			53.1	31.6	13.2	35.6	47.1
(b) baseline on conv5	RPN, C_5	C_5	$2fc$			51.7	28.0	9.6	31.9	43.1
(c) FPN	RPN, P_k	P_k	$2fc$	$\surd$	$\surd$	56.9	33.9	17.8	37.7	45.8

$$k = \left\lfloor k_0 + log_2(\sqrt{wh}/224) \right\rfloor . \quad (1)$$

Here 224 is the canonical ImageNet pre-training size, and k_0 is the target level on which an RoI with $w \times h = 2242$ should be mapped into. Analogous to the ResNet-based Faster R-CNN system [5] that uses C_4 as the single-scale feature map, we set k_0 to 4. Intuitively, Equation 1 means that if the RoI's scale becomes smaller, it should be mapped into a finer resolution level. Based on these adaptations, one can train and test Fast R-CNN on top of the feature pyramid. Implementation details are given in the experimental section.

4. Evaluation of the Model

In the above they used a fixed set of proposals to investigate the detectors. But in a Faster R-CNN system [10], the RPN and Fast R-CNN must use *the same network back-bone* in order to make feature sharing possible. Tab. 1 shows the comparisons between their method and two baselines, all using consistent backbone architectures for RPN and Fast R-CNN. Tab. 1(a) shows their reproduction of the baseline Faster R-CNN system as described in [5]. Under controlled settings, our FPN (Tab. 1(c)) is better than this strong baseline by 2.3 points AP and 3.8 points AP@0.5.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1

[2] R. Girshick. Fast R-CNN. *arXiv preprint arXiv:1504.08083*, 2015. 2

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[4] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014. 2

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 3

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[7] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1

[8] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 1

[9] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016. 1

[10] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 2, 3

[11] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun. OverFeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 1

[12] R. Vaillant, C. Monrocq, and Y. L. Cun. Original approach for the localisation of objects in images. *VISP*, 1994. 1