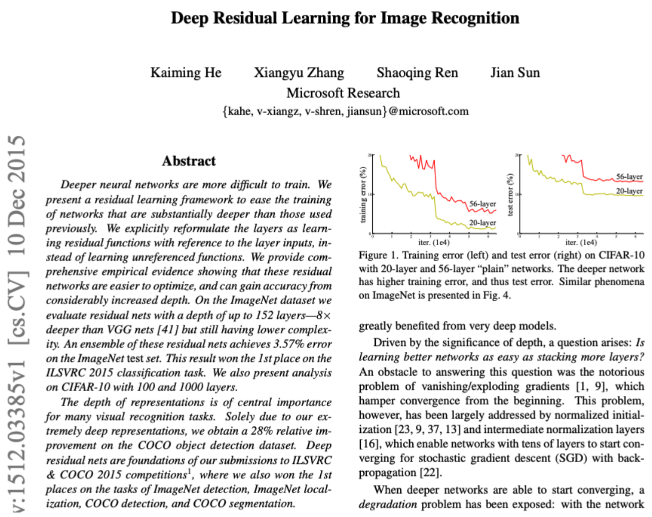
Part 1 : Residual Networks



v:1512.03385v1 [cs.CV] 10 Dec 2015

Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

Abstract

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers—8× deeper than VGG nets [41] but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers.

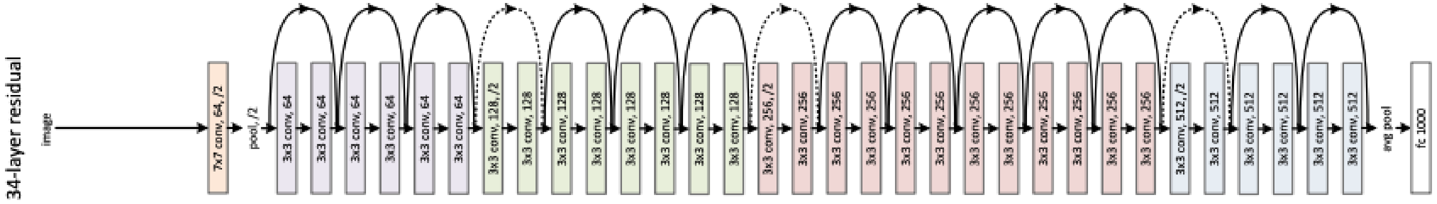
The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions[1], where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.

Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

greatly benefited from very deep models.

Driven by the significance of depth, a question arises: *Is learning better networks as easy as stacking more layers?* An obstacle to answering this question was the notorious problem of vanishing/exploding gradients [1, 9], which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization [23, 9, 37, 13] and intermediate normalization layers [16], which enable networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation [22].

When deeper networks are able to start converging, a *degradation* problem has been exposed: with the network



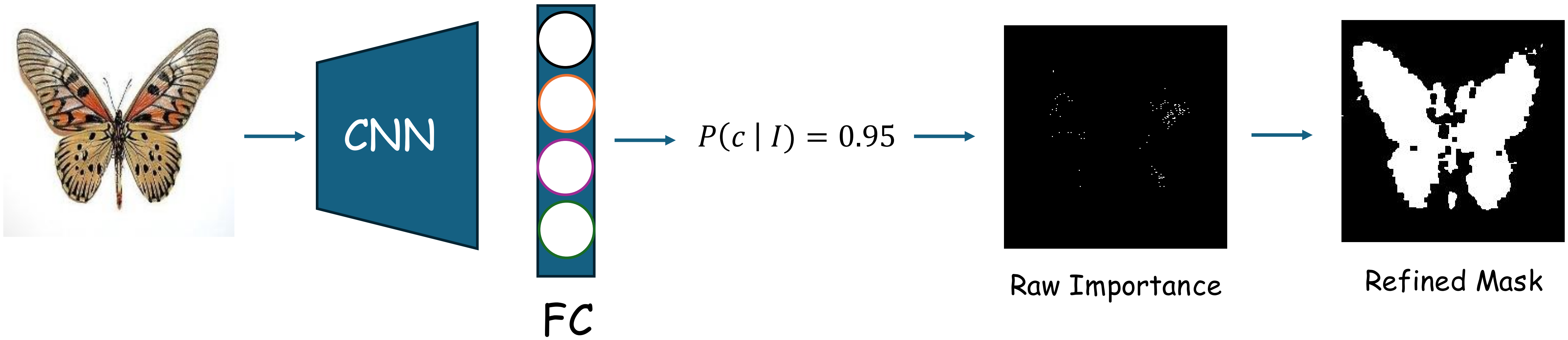
- Read paper : https://arxiv.org/pdf/1512.03385
- Focus on **Fig 3.** and **Sec 3.4**

Dataset

Butterfly and Moth species classification into 100 classes




Part 2 : Free Lunch



- Goal : Given a trained CNN, segment the objects without additional data.

Saliency Visualization

- Given an image I_0, a class c and a CNN with a class score function $S_c(I)$ we'd like to rank the pixels of I_0 based on their influence on the score $S_c(I_0)$

- Considering a linear score model for some class :

$$S_c(I) = w_c^T I + b_c$$

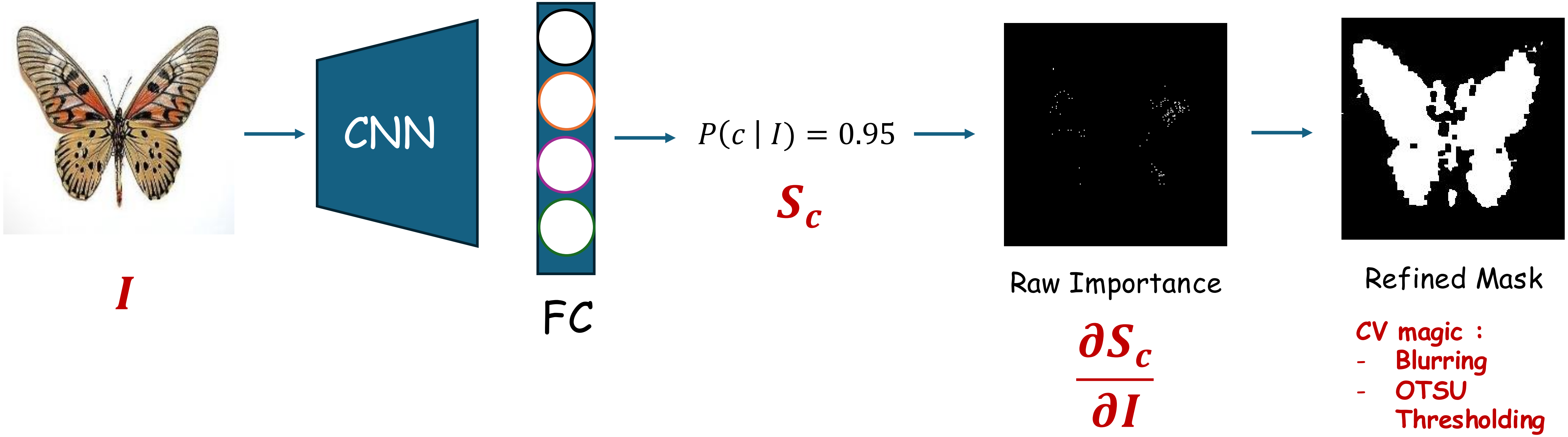
  - Image I is represented in 1-D form and w_c and b_c are respectively the weight and bias vector of the model.
  - Magnitude of elements of w defines the importance of corresponding pixels of I for the class c.

- In CNNs the class $S_c(I)$ is highly non-linear, However we can approximate it with a linear function about I_0 using 1st order Taylor expansion:

$$S_c(w) \approx w^T I + b$$

$$w = \frac{\partial S_c}{\partial I}$$

Revisiting



CV magic :
- Blurring
- OTSU Thresholding
- Dilate/Erode

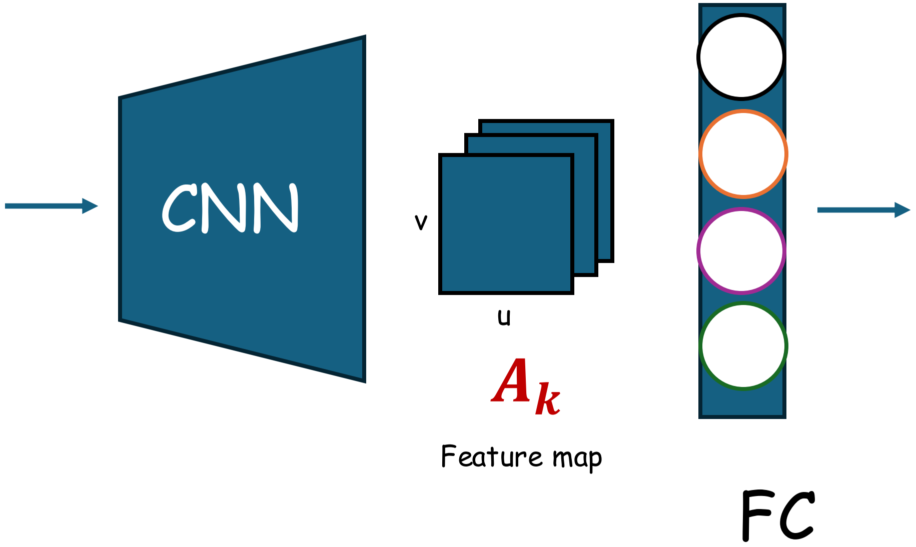
- *Sec 3.2 in [1]*

[1] Simonyan et al, *Deep Inside Convolutional Networks : Visualizing Image Classification Models and Saliency Maps, 2013*

Alternative : Grad-CAM

- Uses the gradient of a 'target concept' flowing into the final convolutional layer to produce a *coarse localization* map highlighting the important regions in the image for predicting the concept.

$\boldsymbol{I}$



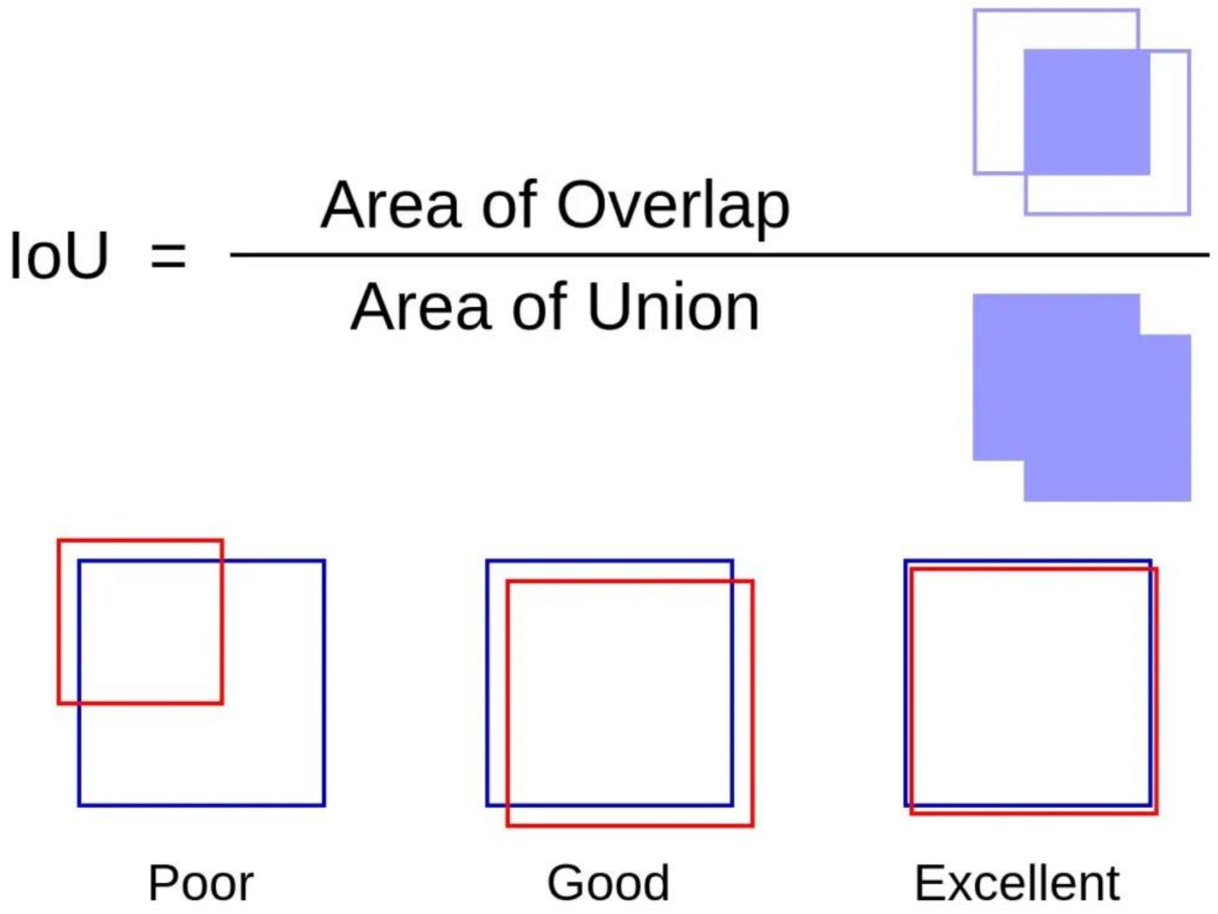
$P(c \mid I) = 0.95$

$\boldsymbol{S_c}$

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial S_c}{\partial A_{ij}^k}$$

$$\boldsymbol{L_{grad-cam} = ReLU\left(\sum_k \alpha_k A^k\right)}$$

Evaluation: IoU Score



References

- Simonyan et al, *Deep Inside Convolutional Networks : Visualizing Image Classification Models and Saliency Maps, 2013*

- Selvaraju et al, *Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, 2016*