Single Image Deblurring and Visual Grounding

A Project Report

submitted by

ADALA MANISH REDDY

in partial fulfilment of the requirements
for the award of the degree of

BACHELOR AND MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2020

THESIS CERTIFICATE

This is to certify that the thesis titled **Single Image Deblurring and Visual Grounding**, submitted by **Adala Manish Reddy**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. A.N. Rajagopalan
Research Guide
Professor
Dept. of Electrical Engineering
IIT Madras, 600 036

Place: Chennai

Date: June 2020

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor **Dr. A.N. Rajagopalan** whose teachings motivated me to pursue my current area of research. I was able to learn a wide range of topics thanks to his guidance. His ideas always intrigued me and led me in new directions helping me expand my horizons. I'm extremely grateful to him for his constant support and guidance.

I would like to thank **Kuldeep Purohit**, PhD student at IPCV lab and **Maitreya**, PhD student at IPCV for their constant support and help in exploring a wide range of ideas.

I would like to thank **Nithin GK**, Dual Degree student at IPCV lab for all brain-storming discussions we had.

I would like to thank **Mahesh Mohan**, PhD student at IPCV lab, for helping me understand and explore conventional methods in Image processing.

I would like to thank my friends **Sarath B, Teja Srikanth, Sandesh, Vardhan, Srinath, Jayakrishna, Srikanth B, Ellesh and Aswath** for being there for me whenever I needed help.

Lastly I would like to thank my parents who have been guiding and helping me at every step of my life. None of this would have been possible without their help.

ABSTRACT

Using deep neural networks for Single Image Deblurring has been well explored in recent years, greatly improving the performance and decreasing the inference time, making real-time deblurring possible. But these networks demand more computational resources as they have a multi-stage architecture, as a result these models have difficulty running on platforms with less computational resources such as mobile phone. We propose a novel training method to improve the performance of small networks providing a light weight alternative for platforms with less computational resources. Our experiments show that we can obtain performance comparable to multi-stage networks by using our training method on light networks. We also look at the problem of restrictive effective receptive field in the context of single image deblurring and propose novel block-wise self-attention block to extract global contextual information and greatly enlarge the receptive field. Our experiments demonstrate the effectiveness of this block whose use can be extended to other Image Processing tasks such as image super-resolution and image denoising.

The problem of visual grounding is widely addressed using a two stage framework where an object detection network gives object proposals and these proposals are then ranked based on language query. We introduce novel feature filtering framework to filter out all the image features that are not being referred to in the language query and predict bounding box from the remaining features. Few other one stage approaches simply append language features to image features. Through our experiments we show that these networks suffer from false positives and that our feature filtering framework partially solves this problem as a result improving the accuracy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

CNN	Convolutional Neural Network
DL	Deep Learning
RNN	Recurrent Neural Network
BWSA	Block Wise Self Attention
SA	Self Attention
MST	Multi Scale Training
MPT	Multi Patch Training
AB	Aggregation Block
DMPHN	Deep Multi-Patch Hier-archical Network
WS	Weight Shared

CHAPTER 1

Introduction

Image deblurring is important image processing task for better visual perception as well as achieving good performance in machine perception or computer vision tasks. Image blurring can be caused by a range of phenomenon such as camera shake, object motion, out-of-focus. It is a highly ill-posed problem and is solved in traditional methods by imposing restrictions on camera motion and/or blur kernel. Most of these methods are optimization based and take way longer compared to their learning-based counterparts.

In recent years learning-based methods have been well explored for the task of Single Image Deblurring and of them networks using multi-stage architectures have been promising [Nah *et al.* (2017), Tao *et al.* (2018)],Zhang *et al.* (2019)]. These work by feeding the image at different scales to the network and learning the latent image stage by stage. The networks are essentially learning the blur information in different stages enforced by the network. In this thesis we propose an alternative method to enforce the network to learn blur information in an incremental manner. We take a network with one scale and train it with images of multiple scales in stages.

In this thesis we explore another problem in the area of Image Deblurring which can potentially be extended to other Image Processing tasks such as Image Super Resolution - the problem of effective receptive field. One reason why multi-scale networks work so well is that it is easier to learn coarser or large blurs when the image is scaled down. Theoretically using deep networks should help solve this problem but practically it is observed that this is not the case as explored by Luo *et al.* (2016). We explore 2D self-attention[Parmar *et al.* (2019)] to help solve this problem.

But self-attention only partially solves the problem as structure of message passing across layers itself leads to restrictive receptive field. Hence, we propose a new block-wise self-attention which considers blocks of image as a single pixels and processes them.

In this thesis we also address the problem of Visual Grounding, which is the task of finding the part of the image being described by the Natural Language input query.

The task of visual grounding is an important step towards human computer interaction. The task can further be categorized into phrase localization, referring expression comprehension, natural language object retrieval. Phrase localization is where the language query is a part of the sentence describing the whole image, there could be multiple such parts. Referring expression comprehension is where the input sentence describes the object to be found in the image. The current work can be applied to both the tasks, in fact it can be applied to any Visual Grounding task.

Whether it is phrase localization or referring expression comprehension [Mao *et al.* (2016*a*), Hu *et al.* (2016), Yu *et al.* (2018)] most of works followed the two stage framework of getting all the object proposals and then ranking them by comparing them with the input language query. On one hand it is highly inefficient to extract features for all the region proposals and checking their similarity with the language query and moreover, it is possible that the object proposals are not good and the region of interest is not present in the proposals. Yang *et al.* (2019) propose a single stage network to overcome these shortcomings, they append features extracted from language query to the visual features and pass the resulting feature map through object detection network.

Appending language features to the visual features to include the query information is sub-optimal, in the current work we explore feature filtering framework to filter visual features while looking at the language features.

Next we're going briefly discuss some papers on Image Deblurring that use multi-scale approaches and some works that extend self-attention to images.

1.1 Deblurring using Neural Networks

In this thesis we deal with motion blur which is caused due to camera motion or motion of objects in the scene. This can also be attributed to the comparatively high exposure time. Not only does this make the photo look bad but also makes computer vision applications such Object Detection and Image Classification very difficult to work. A popular way to model motion blur is

$$B = KS + n$$

where B,S are vectorized blurry and latent images, rows of K act as blur kernel and n is noise.

If we already have information about the kernel the problem of finding the latent image is called non-blind deblurring and has been well explored. But in general that not the case and the only thing we have is the blurred image. The problem of finding latent image without knowing the kernel information is called blind deblurring. This is an ill-posed problem hence, some approaches were to parameterize blur by making assumptions on its formation such as only camera rotation. But these approximations are not accurate and hence we look to neural networks to learn complex blur kernels from data.

In the area of learning based methods we are interested in the multi-scale approaches in which sub-sampled image or part of the image is fed to the network at multiple stages. In the next section we are going to look at few multi-stage approaches.

1.2 Multi-stage deblurring

Paper summary: Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring

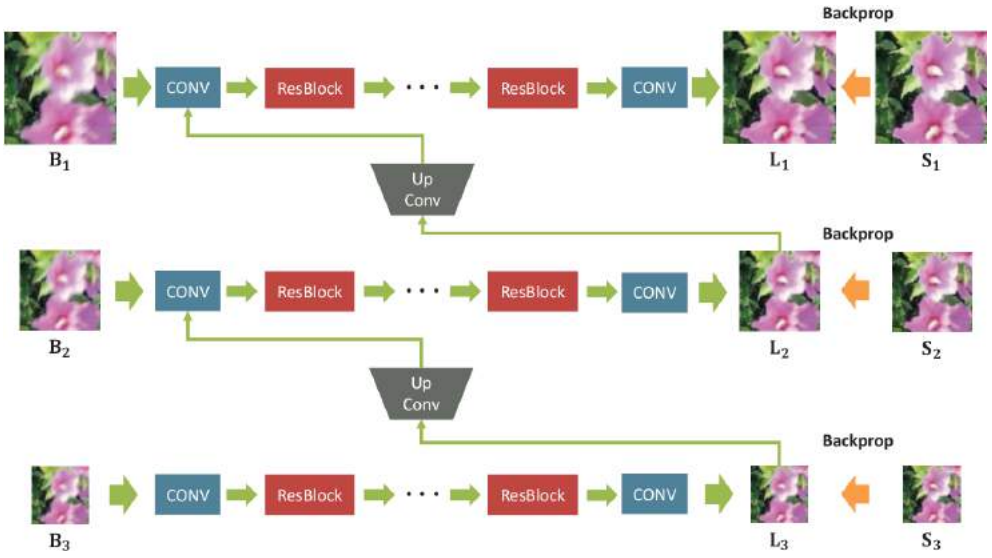

Figure 1.1: Network architecture taken from [Nah *et al.* (2017)]

In this paper paper the authors [Nah *et al.* (2017)] propose a multi-scale end to end

trainable fully convolutional neural network. As shown in figure 1.1 the image is down sampled and fed to the network at multiple stages. The authors use residual network as building block after removing all the batch normalization layers and ReLU layer at the output. The idea behind using multiple scales is to make the learning process easier by having the first stage coarsest blur kernel and next stage learn finer blur and so on.

The authors use two types of losses - content loss which is mean squared error between deblurred image and latent sharp image and adversarial loss from the discriminator which classifies an image as deblurred or sharp one. The net loss is as follows:

$$L_{total} = L_{cont} + \lambda * L_{adv}$$

where $\lambda = 10^{-4}$

They also introduce a new dataset called the GOPRO dataset consisting of 3214 blur and image pairs at resolution of 1280x720. They take videos at 240fps and average (7-13) latent frames to get a blurred image and sharp image is taken to be the mid-frame.

Paper summary: Scale-recurrent Network for Deep Image Deblurring

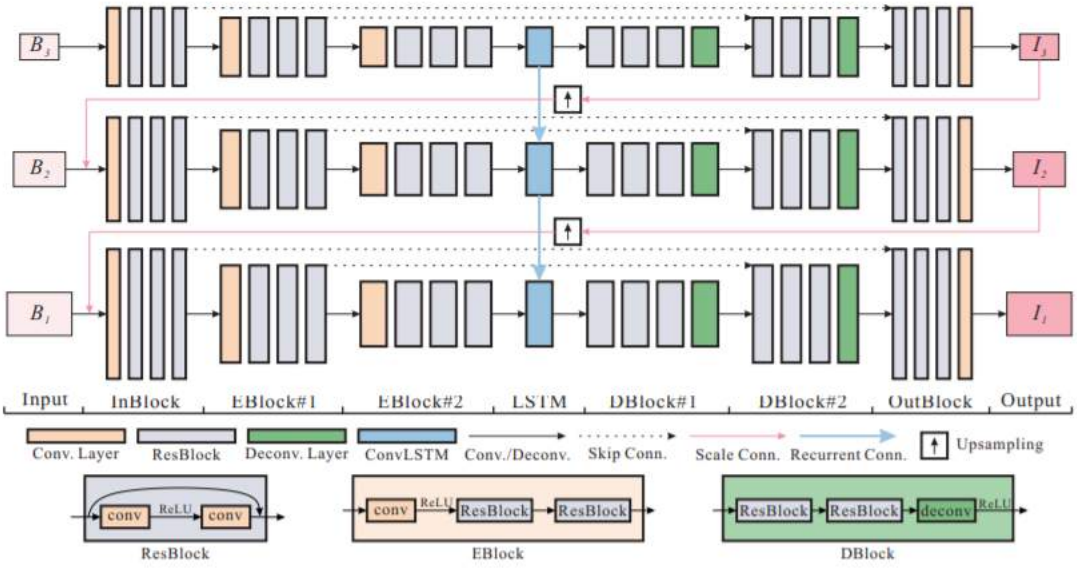

Figure 1.2: Network architecture taken from [Tao *et al.* (2018)]

The authors [Tao *et al.* (2018)] of this paper take the multi-scale approach introduced by Nah *et al.* (2017) a step forward. They suggest that we do not need to learn

different functions at different scales by pointing that the problem we are trying to solve is the same. Hence, they propose a recurrent structure as shown in figure 1.2.

The recurrent structure helps reduce the number of parameters by a scale of three. It is interesting to note that network with recurrent structure performs better compared to the original three scale network which is attributed to instability or unrestrictive solution space by the authors. The output at scale i is calculated as:

$$I^i, h^i = Net_{SR}(B^i, I^{i+1\uparrow}, h^{i+1\uparrow}; \theta_{SR})$$

where B^i, I^i are the input, output at that scale, Net_{SR} is the proposed network , θ_{SR} the training parameters, and the hidden state h^i flows through layers. The authors also add skip connections between encoder and decoder inspired by their effectiveness in other restoration tasks [Mao *et al.* (2016*b*)]. They use sum of mean squared error at all scales as loss function.

Paper summary: Deep Stacked Hierarchical Multi-patch Network for Image Deblurring

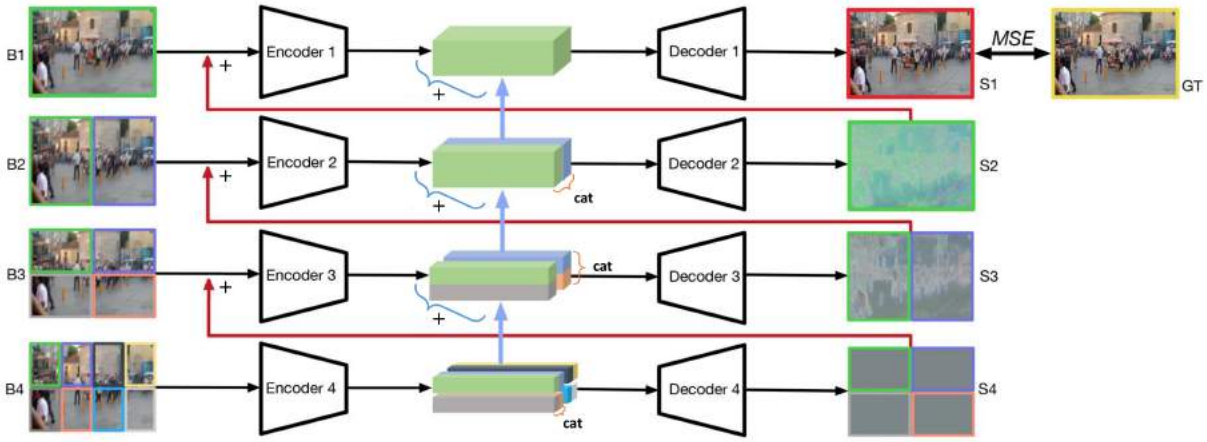

Figure 1.3: Network architecture taken from [Zhang *et al.* (2019)]

The improvement in performance with increase in the number of scale saturates at around 4 scales. Increasing the depth of the network also does not help improve the performance. Hence, in this paper the authors propose dividing the image into parts, deblurring them and sending them to next stage after concatenating them as shown in the figure 1.3. The architecture used here is similar but shallower compared to the

previous networks.

In this setup the level that deals with smallest patches learns finest blur kernel and the next layer learns coarser blur kernel and so on. In this way the hierarchy is reversed compared to multi-scale approaches where coarsest blur kernel is learned first (in terms of network levels). The loss is mean squared error between the final deblurred image and the ground truth.

The authors also introduce a novel stacking approaches and show that they do not saturate as in the case of multi-scale approaches. The current network has inference times 40 times faster than the previous multi-scale approach. This is a result of shallow network with small kernel sizes and lack of need for up-sampling between different levels.

1.3 Self-Attention in Images

The self-attention module has been introduced in the field of language modelling [Vaswani *et al.* (2017)] to better capture long range dependencies and parallelize training and inference. Convolutional Networks have been have used for extracting features from images for a long time, yet they face difficulty in capturing long range dependencies. The residual networks [He *et al.* (2016)] address this problem by enabling us to train deeper networks. But increase the depth of the network is an inefficient way to capture long range dependencies as shown by Luo *et al.* (2016).

Recently, there have efforts to use content based interactions to better capture image features. One approach is where channel-wise attention [Hu *et al.* (2018)] is introduced to capture inter-dependencies between channels. Non-local Neural networks were introduced by Wang *et al.* (2018) to capture long range dependencies spatially. In this architecture each pixel (or spatial position) is calculated with attention over all the pixels in the previous layer. This a computationally expensive, hence the feature map is significantly down-sampled. To reduce the computational complexity self-attention is modified to take attention over neighbouring pixels only as shown in 1.4.

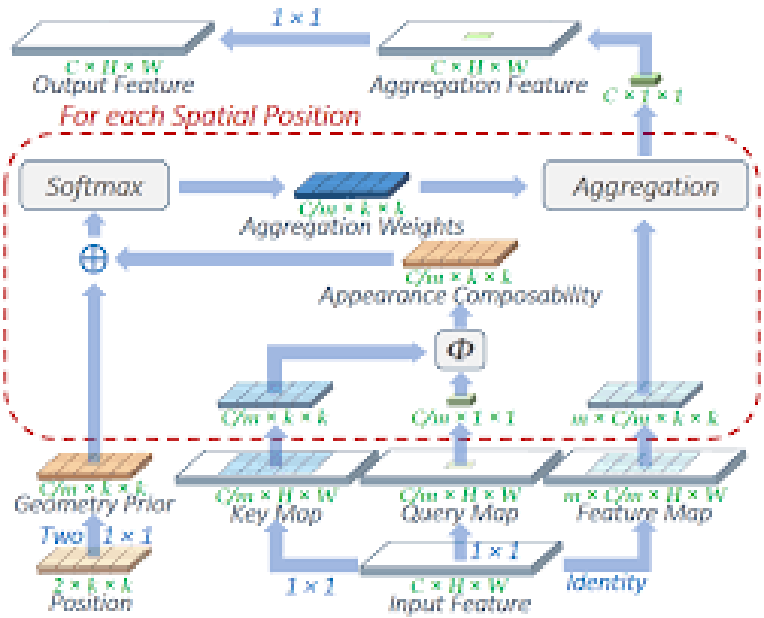

Figure 1.4: Network architecture taken from [Hu *et al.* (2019)]

Paper summary: Stand-Alone Self-Attention in Vision Models

The approaches we've discussed above use the attention layers as an augmentation to convolutional layers [Bello *et al.* (2019)]. In the current paper [Parmar *et al.* (2019)] the authors take this a step forward to using only attention layers to extract features from images. To demonstrate the effectiveness of extracting features using attention layers they are used in core Computer Vision tasks - Image Classification and Object Detection.

Another important feature of attention layers is that the number of parameters does not scale with the kernel size making it more parameter efficient. As a result, competitive performance is achieved with much fewer parameters - ImageNet classification with 29% fewer parameters and COCO object detection with 34% fewer parameters.

It is to be noted that adding positional information is an important part of self-attention as otherwise all the neighboring elements are treated in the same way and all positional information in the image is lost. This is verified through ablation studies where removing the positional information reduces the accuracy of Object Detection by 2%. There are many ways to add positional information - adding features corresponding to width and height as in Vaswani *et al.* (2017) or using geometry prior as in Hu *et al.* (2019). In the current paper the relative positional features for each position are learnt.

Figure 1.5: Relative distance computation- taken from [Parmar *et al.* (2019)]

Next we are going to summarize some object detection networks that we use in our architecture for Visual Grounding. In particular we are going to look at the YOLO object detection networks.

1.4 YOLO Object Detection

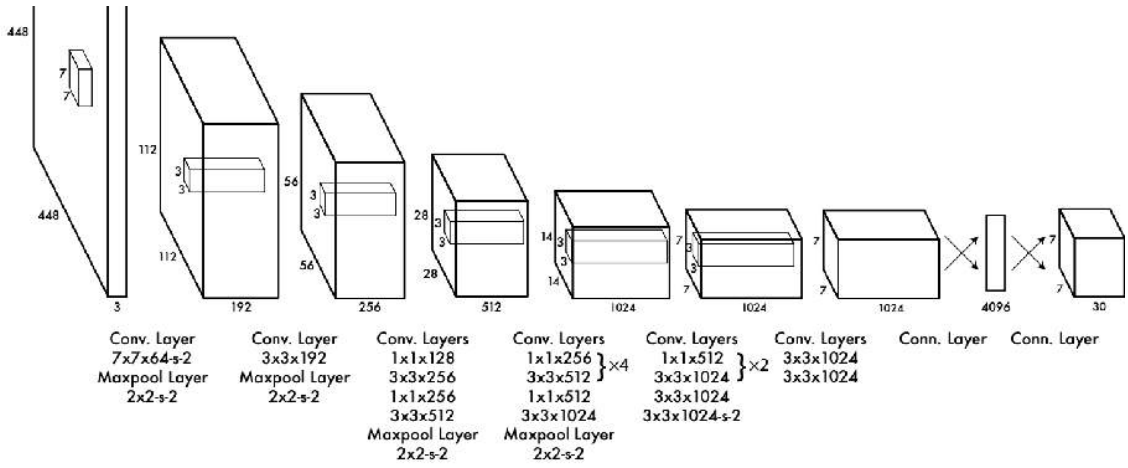

Figure 1.6: Yolo detection network - Redmon *et al.* (2016)

Redmon *et al.* (2016) introduced a novel approach for object detection when the widely used framework was to get region proposals and predict whether each proposal was an object or not and the class of the object. Yolo is extremely fast as the whole image is processed at once rather than processing each proposal at a time.

The framework introduced by Yolo is to divide the image into several grids and for

each grid predict B bounding boxes and the confidence scores for each bounding box. The confidence score of a bounding box represents the percentage of overlap it has with a ground truth object. Whenever there exists an object the confidence score equals the IOU (intersection over union) and is zero otherwise. For each bounding box we predict (x,y) offset of the center of bounding box from the grid, (w,h) width and height of the bounding box and confidence score of the bounding box. For each bounding box C class conditionals are also predicted. A class conditional is the probability that the object enclosed is of that particular class given it is an object.

$$Pr(class_i|Object) * Pr(Object) * IOU_{pred}^{truth} = Pr(class_i) * IOU_{pred}^{truth}$$

Redmon *et al.* (2016) use grid size of 7x7 and for each grid cell they predict 2 bounding boxes and the number of classes is 20. The network structure used is as shown in 1.6.

1.4.1 Version 2

Redmon and Farhadi (2016) make a bunch of modifications the Yolo framework to further improve the performance. The pre-training of the network on the task of image classification improves the performance on detection. The authors add batch normalization [Ioffe and Szegedy (2015)] and pre-train on high resolution images which moderately improves the performance.

An important modification is the use of anchor boxes to predict bounding boxes instead of directly predicting the bounding boxes. The anchors boxes are priors that are picked and fixed before the training starts. Then for each anchor box the center offset, width, height offset, confidence and class conditional are predicted. The coordinates of the bounding boxes are calculated as follows:

$$b_x = \sigma(t_x) + c_x$$

$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w e^{t_w}$$

$$b_h = p_h e^{t_h}$$

$$confidence = \sigma(t_o)$$

where t_x, t_y, t_w, t_h and t_o are predicted by the network and c_x, c_y are coordinates of top left corner of prior and p_w, p_h are width and height of the prior.

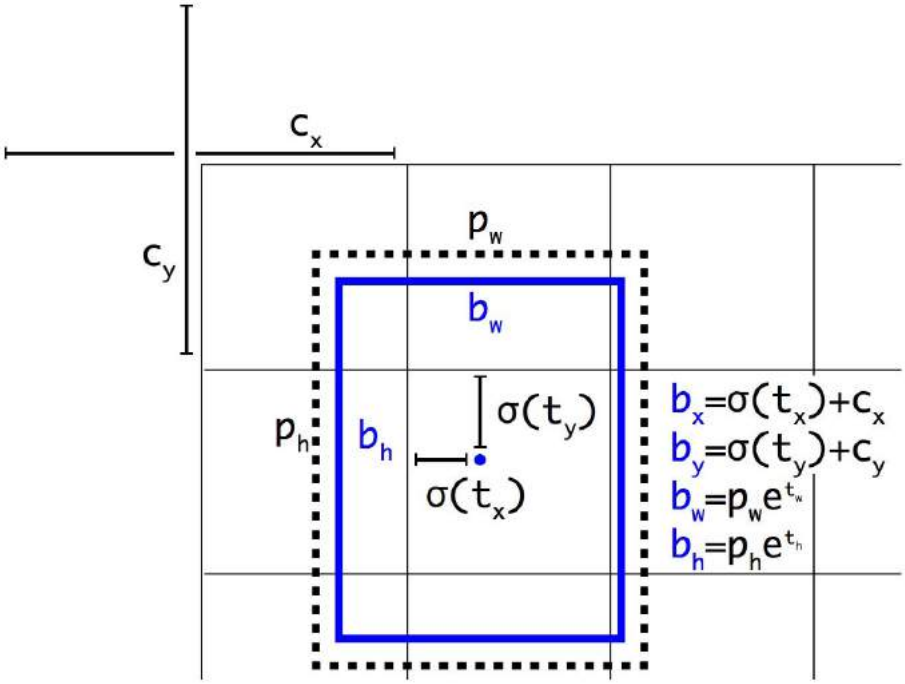

Figure 1.7: Bounding box with offset prediction and prior - Redmon and Farhadi (2016)

1.4.2 Version 3

In version 3 Redmon and Farhadi (2018), the authors incorporate feature extraction and prediction of boxes across three different scales. The back-bone network is larger - 53 convolutional layers whereas the one used in version 2 has 19 convolutional layers. Three bounding box priors are used for each scale which makes a total of 9 priors. Instead of using softmax independent logistic classifiers are used to address overlap of object classes. In our network we use this version of Yolo as the back-bone architecture.

CHAPTER 2

Theory Background

In this section we will look into some of the basic building blocks and structures used in learning based methods to solve Image Processing and Computer Vision problems.

2.1 Convolutional Neural Network

Convolution Neural Networks have been extensively used in Image Processing and Computer Vision for feature extraction. Their effectiveness was first proved by Krizhevsky *et al.* (2012) in task of Image Classification by demonstrating considerable improvement over existing methods.

It is computationally very expensive to use Feed Forward Neural Networks to process images, another problem could be the large number of parameters which could be hard to train. Hence we look to a low cost alternative - Convolutional Networks - used to exploit the local dependency in images.

2.2 Residual Network

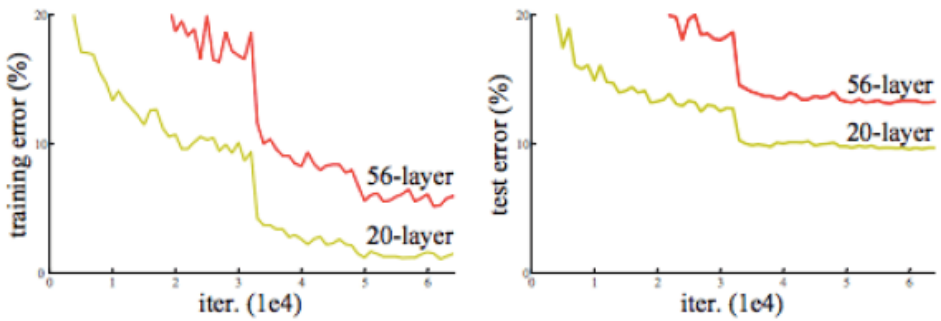

Figure 2.1: Training error and testing CIFAR-10 with networks of different depth. Figure from:-He *et al.* (2016)

The performance over computer vision tasks was improved by increasing the depth of networks and altering the kernel size. But, after reaching a certain point training deep networks becomes a difficult task as demonstrated by He *et al.* (2016).

To solve this problem they introduce shortcut links in the network to help easier learning. The residual block used is as defined below. The intuition behind this is that the deeper stages of the network will only have to learn difference between actual function and function already learnt by the previous layers.

$$y = \mathcal{F}(x, W_i) + x$$

where x,y are input and output and W_i are parameters to be learned.

2.3 Encoder-Decoder Skip Connections

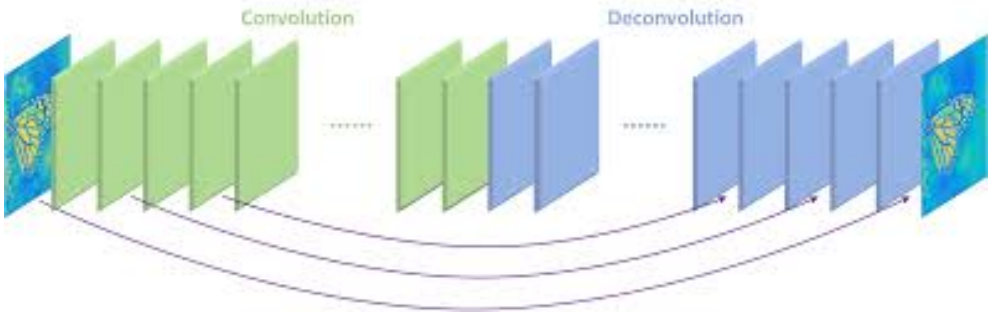

Figure 2.2: Skip connections - [Mao *et al.* (2016*b*)]

The encoder-decoder fully convolutional architectures have been used for Image Restoration tasks to deal with images of varying sizes and since they work well in practice. But as the networks get deeper and the representation at the end of encoder get smaller (too small in dimension) information loss is expected.

This idea was first used for Image Segmentation where Ronneberger *et al.* (2015) introduce the U-Net Architecture. The idea is to compensate for the information loss by adding direct skip connections between encoder and decoder. These skip connections can be implemented as adding encoder and decoder features or appending them. This idea has since been widely used in Image Restoration tasks such as Image Denoising [Mao *et al.* (2016*b*)] and Image Deblurring [Nah *et al.* (2017)].

2.4 Self-Attention

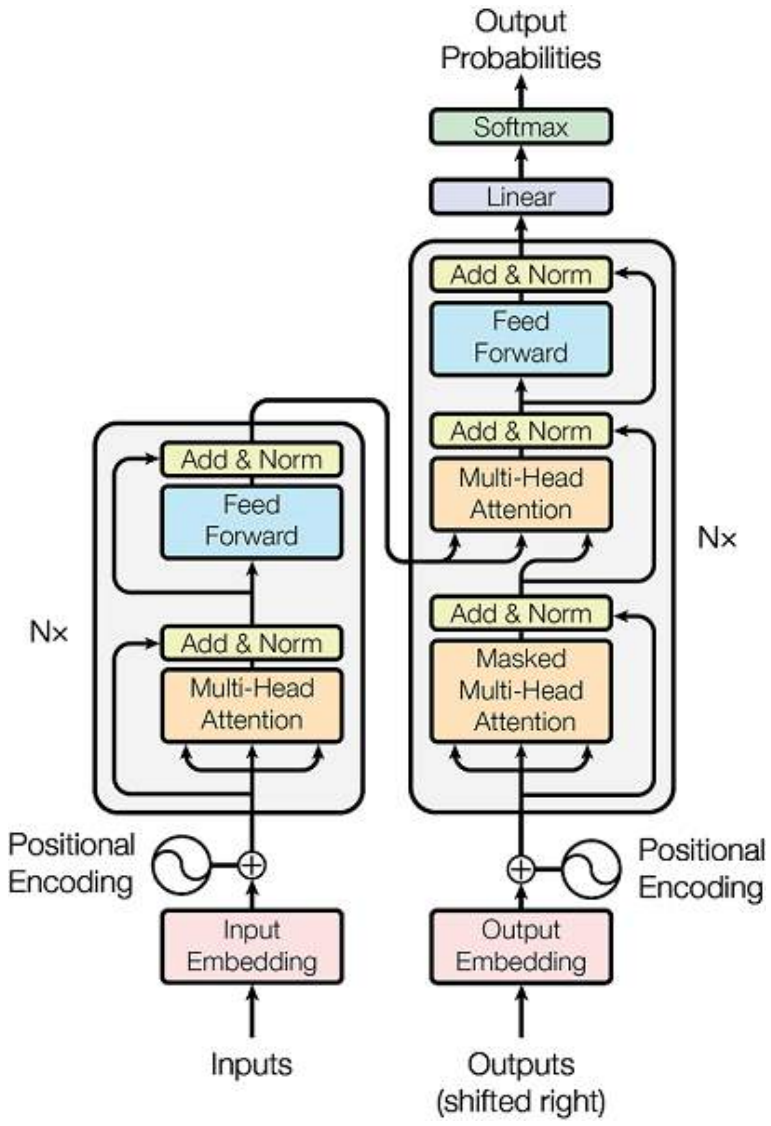

Figure 2.3: Encoder-decoder architecture - [Vaswani *et al.* (2017)]

Self-Attention was first introduced in the area of Language processing by Vaswani *et al.* (2017) as an alternative to Recurrent Neural Networks. The idea was to parallelize language processing in order reduce training time and also address vanishing and exploding gradient problems in Recurrent Neural Networks. This idea is an extension of attention between encoder and decoder in language understanding tasks. The architecture used for language modelling is as shown in 2.3.

In self-attention each feature in the next level is calculated with attention over every feature in the previous layer. Moreover, dot-product attention is used here instead of additive attention. From each feature we calculate three features - query, key and value

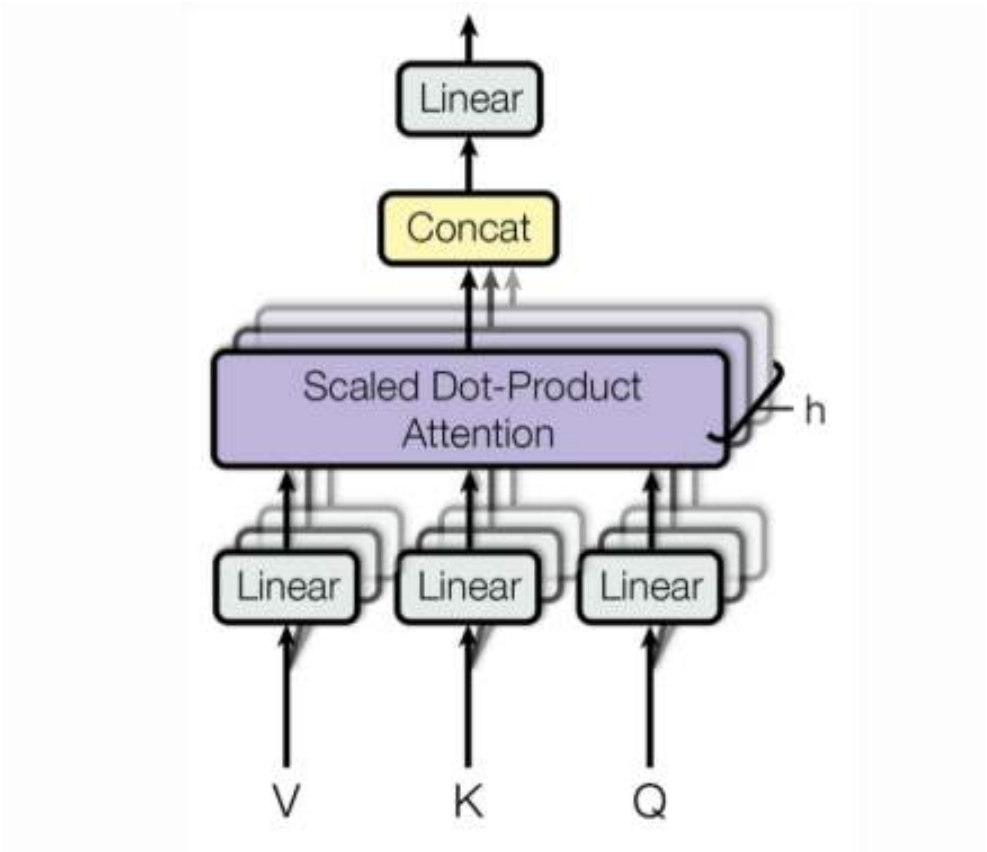

Figure 2.4: Multi-head attention - [Vaswani *et al.* (2017)]

- and use dot-product between key and query of different features to score their compatibility or relation. We take weighted sum of all the values as our output feature vector where weights are obtained by taking dot-product of query of current feature and keys of all the features. In practice it is calculated as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

where Q is matrix stacked with query vectors, K is matrix stacked with key vectors, V is matrix stacked with value vectors and d_k is a dimension of keys.

Multi-head attention is to run multiple such attentions in parallel and append all the output features in the end as show in 2.4. Having multiple heads enables each of the head to focus on different aspects of the feature all the while maintaining the same computational cost as we reduce feature sizes by a scale of number of heads.

CHAPTER 3

Multi-stage Training

In this chapter we propose and study multi-stage training for single image deblurring as an alternative to multi-stage networks. In the previous chapter we've looked at few multi-stage networks [Nah *et al.* (2017), Tao *et al.* (2018), Zhang *et al.* (2019)] used for Single Image Deblurring. The multi-stage structure helps the networks train easily. We are interested in the weight shared model where number of parameters is same as the that of a single stage but is better in terms of performance. This is due restriction imposed on learning by the structure of the network. We try to use this idea on a single stage network and restrict the learning process by training it in a multi-stage manner. This gives comparable performance to multi-stage network while giving dramatic inference speed gain.

3.1 Motivation

We are going to look at the Multi-patch network [Zhang *et al.* (2019)] we've seen in the previous chapter to motivate our proposal for multi-stage training. In the multi-patch network the image is divided into parts and each part is passed through the encoder independently at each stage and then the parts are concatenated at the end of encoder before the decoder.

The current network is a fully convolutional network hence, it shouldn't matter if parts of the image are passed through the convolutional network or the whole image is passed at once. There would only be differences at the boundaries which is due to zero padding. We confirm this by inputting a blurry image of size 720x1280 in both the ways and finding the difference between outputs as shown in 3.1. First we input a blurry image part after part as described in [Zhang *et al.* (2019)] to get De-blurred Image1 in 3.1. Then we send the whole image as input at all the stages of the network to get De-blurred Image2 in 3.1. The difference between both the deblurred images is negligible hence, we scale any non-zero values to one (max value) to visualize even the

Figure 3.1: Output of the Multi-patch network

slightest differences shown in 3.1. The current experiment is performed for architecture with three scales (1-2-4).

However, sending whole image as input during training gives comparatively worse performance. It is to be noted that during training the input image is of size 256x256 and at scale four each part of the image is of size 64x64. The difference at boundaries can be as thick as the receptive field and hence for a small image there could be a great difference between both methods of input. Having a small image as input helps the network learn finer kernels. We conclude that the input method described in [Zhang *et al.* (2019)] helps train the network better and does not have any structural advantage inherently.

Hence, we conclude that we can obtain comparable performance by constraining the learning process similarly by introducing a multi-stage training process.

3.2 Multi-Patch Training

The Multi-Patch training (MPT) process is inspired from the network structure in Zhang *et al.* (2019). As this is just a training algorithm this can implemented for any network

structure. In this algorithm at each stage of training the image is divided into a certain number of parts and each part is sent through the network. The output is obtained by spatially concatenating outputs of all the parts.

$$I_i = Net(P_i)$$

$$I = Concat(I_1, .., I_i, .., I_n)$$

where each P_i is a part of blurry image B_i and i $\in [1, n]$. n is the number of parts the image is divided into.

The network is initially trained by dividing the image into large number of parts (8 for example) and once the training saturates we decrease the number of parts the image is divided into. This process is continued until we complete training with the number of parts as one. This way the network first learns fine blur kernels and learns coarser blur kernels with each stage.

It is important to note that the size of input should be small (256x256) during the training process for this algorithm to give good performance. As the small size of the image is what helps the network to learn finer blur kernels.

3.3 Multi-Scale Training

The Multi-Scale Training (MST) process is similar to the Multi-Patch Training process but the blur is learned in coarser to finer manner instead as in Tao *et al.* (2018). In this algorithm the image is down-scaled and given as input to the network. Once the training a particular scale stabilises the scale of the image is increased. This process is repeated until the network is trained with image of original size.

The reverse of this algorithm would be to start training with the original sized image and then down-scale the image once that training saturates. Repeating this process for required number of stages. The reverse of Multi-Scale Training can added onto Multi-Patch Training. The resultant algorithm would be to train in Multi-Patch fashion and once single part (or full image) stage is reached continue onto reverse Multi-Scale training. In this way the network is trained from the finer extreme to the coarser extreme.

CHAPTER 4

Block-Wise Self Attention

In this chapter we review effective receptive field in the context of single image deblurring and propose a novel block-wise self-attention to extract global contextual information and expand receptive field. The problem of Single Image Deblurring is ill-posed. The resultant blurry image could've formed as a result of a wide range of blur kernels of multiple scales. For example motion of an object close to the camera would result in a small blur whereas motion of an object farther from the camera would result in a large blur. To capture large blurs we need a network with large receptive field. Adding depth to the network helps to some extent but is not effective as the effective receptive field is only a fraction of theoretical receptive field as shown by Luo *et al.* (2016). The multi-scale networks and the multi-scale training we've looked at in the previous section increase the effective receptive field forcibly through the training process. In this section we are going to look at content based feature extraction and a novel block-wise feature extraction to enhance the effective receptive field.

4.1 Motivation

Each unit in convolutional networks depends only on a region of input and this region is called the receptive field. It is important that we have an idea about the receptive field of a network as a pixel outside the receptive field cannot affect the output and we are missing out on any information outside the receptive field. Hence having large receptive helps computer vision tasks such as Image Classification where the object to be classified could be as large as the image. Increasing depth is one of the straight-forward ways to increase the receptive field, as a result the depth of architectures for image classification has increased over the years - 5 layers AlexNet [Krizhevsky *et al.* (2012)], 19 layers VGG Network [Simonyan and Zisserman (2014)] and 22 layers GoogleNet [Szegedy *et al.* (2015)].

However, it can be seen that center of the receptive field has more number of paths to the output compared to pixels at the border of the receptive field. As a result center of the receptive has more impact on the output, Luo *et al.* (2016) explore this intuition further and introduce the notion of effective receptive field. They study the distribution of impact in the receptive field and conclude that in most cases it distributes as a Gaussian centered at the center of receptive field. The measure of impact used is the partial derivative of the output unit with respect to the particular pixel in the receptive field.

$$Measure\ of\ impact = \frac{\partial y_{0,0}}{\partial x_{i,j}}$$

where $y_{0,0}$ is the (0,0) unit in the output and $x_{i,j}$ is the (i,j) pixel of input.

Any point inside receptive field having an impact greater than 95.45% of the center point is considered effective and the size of effective receptive field (ERF) is taken as the square root of number of effective pixels.

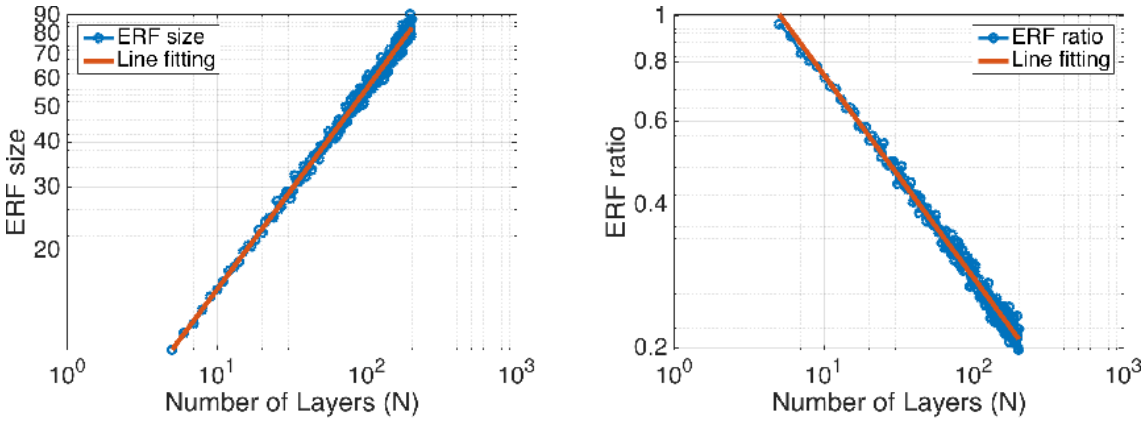

Figure 4.1: Absolute growth (left) and relative shrinkage (right) of ERF - Luo *et al.* (2016)

Luo *et al.* (2016) show that effective receptive field grows linearly with square root of the number of layers (or depth) of the network. And ratio of effective receptive field to theoretical receptive field decays linearly with inverse of the square root of the depth of the network. So, it not an efficient strategy to increase the depth of the network to increase the receptive field.

Hence, to increase the receptive with less increase in computational cost we study content based feature extractors for the task of Image Deblurring which proved effective for Computer Vision tasks image classification and object recognition. We also

introduce a new block-wise convolution structure to address the same problem.

4.2 Network Architecture

The network architecture we use is very simple. We take the Multi-Patch [Zhang *et al.* (2019)] encoder-decoder network as the back-bone as it is simple without any encoder-decoder skip connections and is easy evaluate.

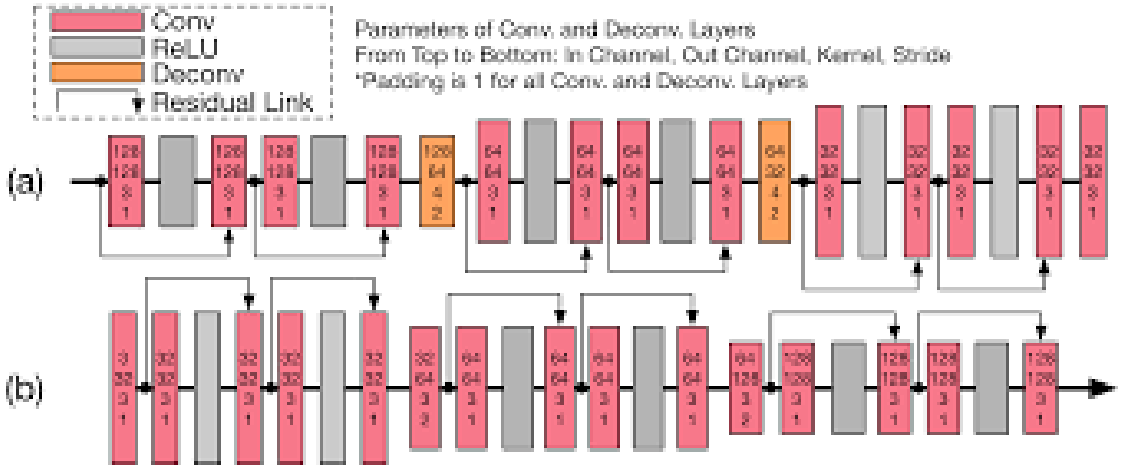

Figure 4.2: Encoder(b)-Decoder(a) Architecture - Zhang *et al.* (2019)

To increase the receptive field we add an Aggregation Block (AB) at the end of encoder. The Aggregation Block helps in extracting the global context, adding these non-local features to the encoded representation helps the network in better deblurring the image. The integration of global context to local information is important to capture large blurs. Capturing global context also helps when different parts of the image experience similar blur or when the blur across the image is related as is in the case of blur due to camera motion. The Aggregation Block in effect tries to capture inter-dependencies in blur across the image.

$$X = Encoder(B)$$

$$Y = AB(X)$$

$$I = Decoder(Y)$$

where B is the blurry input image, AB is the Aggregation Block and I is the deblurred

output image.

4.3 Aggregation Block

The aggregation block is implemented using the self-attention mechanism similar to the one seen in chapter 1. We divide the feature map into non-overlapping blocks of given size. For this the width and height of the feature map have to be multiples of block size, so we pad the feature map appropriately (symmetrically along as possible). Now, each block is treated like a single unit while applying self-attention, hence the kernel size and the stride also have to be a multiple of the block size.

Now, we calculate compatibility between blocks and take weighted sum of all the neighbouring blocks depending on the kernel size to get the output block. For each block we compute query and key feature vectors. The dot-product between key and query of two blocks is the score of their compatibility.

$$k_{ij} = W_k \cdot X_{ij}$$

$$q_{ij} = W_q \cdot X_{ij}$$

where X_{ij} is the ij^{th} block of input feature map X, k_i, q_i are the key and query for ij^{th} block and W_k, W_q are learn-able parameters. The calculation of keys and queries can be optimized by implementing it as two convolutions with kernel size and stride equal to block size and out channels equal to key and query dimension (d_k).

It is important to add spatial information to distinguish between all the neighbouring blocks. We use relative attention as defined in Parmar *et al.* (2019) to add spatial information. The features for each relative position are learned and are for dimension $d_k/2$ so that we can append relative row positional feature and relative column positional feature to get a d_k dimensional relative positional feature. The output of block-wise self-attention (BWSA) is calculated as

$$Y_{ij} = \sum_{a,b \in Nbrs(X_{ij})} softmax_{ab}(q_{ij}^T k_{ab} + q_{ij}^T r_{a-i,b-j}) X_{ab}$$

where $r_{a-i,b-j}$ is relative positional feature and $Nbrs(X_{ij})$ represents neighbours of X_{ij} which is determined by the kernel size. The value block is not calculated through linear transformation as this would be expensive both computationally and in terms of number of parameters.

The feature map obtained as output of block-wise self-attention is then cropped to match the size of input X. Then X and Y are concatenated across channels which would result in doubling the number of channels. We then pass it through a convolutional layer of kernel size 1 with number of out-channels equal to the number of channels of X. This is the final output of the aggregation block.

$$Y = BWSA(X)$$

$$Output = CNN(Concat(X,\ Crop(Y))$$

where X is the input of the Aggregation Block, BWSA is block-wise self-attention operation defined above, CNN, Concat, Crop operations are as described in the previous paragraph and 'Output' is the output of the Aggregation Block.

Why block-wise over pixel-wise? Firstly it is computationally more efficient for a given kernel size. Moreover, when used in Language Modelling each word has a meaning by itself and we are modelling relations between words but in images a single pixel in the feature map might not have a meaning by itself. On the other hand a block of pixels may have contextual meaning by itself, hence we model relations between blocks.

CHAPTER 5

Attentive Filtering

5.1 Motivation

In this chapter we propose a new feature filtering method to suppress false positives in visual grounding. In the current work we propose a framework where we filter visual features extracted from the image depending on whether they are described in the language query. The filtered features are then passed through an object detection network to predict the bounding box. We suggest this as an alternative to the two-stage framework where all the objects are detected first and then ranked based on their similarity with the language query. Yang *et al.* (2019) have a framework similar to ours but they append the language features to the visual features and pass the resulting features through object detection network.

5.2 Network Architecture

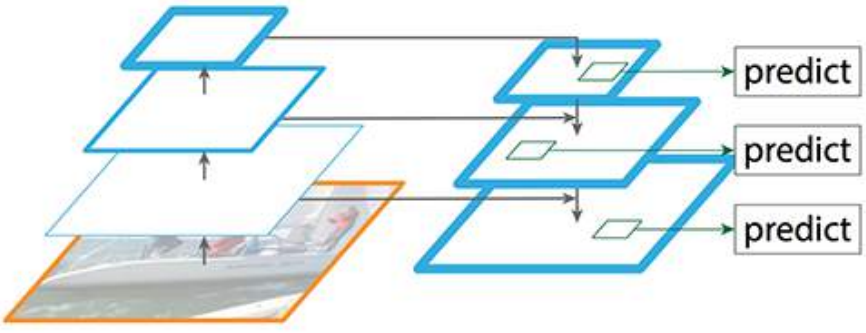

Figure 5.1: Feature Pyramid Network - Lin *et al.* (2017)

We use the Darknet-53 [Redmon and Farhadi (2018)] pre-trained on COCO object detection [Lin *et al.* (2014)] as our back-bone to extract visual features. The Darknet-53 is based on feature pyramid network which is used to extract visual features at different scales, it is shown to improve performance of object detection networks by Lin *et al.*

(2017). The visual features are extracted at three different scales and are of different sizes. When the language query contains location based details it is important to add positional embeddings. We add simple position embeddings similar to Yang *et al.* (2019) which are as follows:

$$(\frac{i}{W'}, \frac{j}{H'}, \frac{i+0.5}{W'}, \frac{j+0.5}{H'}, \frac{i+1}{W'}, \frac{j+1}{H'}, \frac{1}{W'}, \frac{1}{H'})$$

The language features are encoded using an bidirectional LSTM. Then we filter visual features at each scale using dot-product attention over language embedding. The language embedding acts as query and each position in the visual feature map is mapped to a key and a value. The weights of each position are calculated by applying softmax over compatibility values (query-key dot-product) of each position.

$$q = W_q * L$$

$$key_{ij}^k = W_{key}^k * V_{ij}^k$$

$$value_{ij}^k = W_{value}^k * V_{ij}^k$$

$$FF_{ij}^k = softmax_{ij}(q^T key_{ij}^k) value_{ij}^k$$

where k$\in \{1, 2, 3\}$ is the scale of the visual features, L is language query embedding, q is the query, V represents visual features and FF represents the Filtered Features. W_{key}^k and W_{value}^k are learn-able parameters. And $softmax_{ij}$ denotes softmax applied over all positions (ij) of that particular scale.

The equations are written for single head attention for simplicity, but our experiments show that using multi-head attention gives better performance. The reason could be that the task of attention is now divided and each head can pay attention to a particular detail. We try different ways of using the filtered features such adding them to the original features, appending them to the original features and we also try adding batch normalization. We find that passing the filtered features without any modification gives better performance and adding batch normalization [Ioffe and Szegedy (2015)] speeds up training and improves accuracy by 0.5%.

The filtered features are then passed to the object detection network which then

predicts center offset, width offset, height offset and confidence score for each bounding box prior. Note that we don't need the class conditionals here.

CHAPTER 6

Experiments

6.1 Quantitative Results on Multi-stage Training for Image Deblurring

All our experiments are done on NVIDIA RTX 2080 TI GPU. We use the GoPro dataset [Nah *et al.* (2017)] for all our experiments. There are a total of 3214 blur and sharp image pairs of resolution 720x1280 in this dataset of which we use 2103 image pairs for training and remaining 1111 pairs for testing. We train all the networks on 256x256 patch randomly cropped from each image. We use batch size of 8 during training for all the results reported.

To have a fair comparison all the models reported below have same number of parameters. The current experiments are performed with Deep Stacked Multi-patch Hierarchical Network (DMPHN) architecture and the DMPHN models with multiple stages are weight shared (WS).

Table 6.1: Quantitative results for multi-stage training on GoPro dataset

Model	PSNR	SSIM	Runtime(ms)
DMPHN(1)	28.76	0.9134	7
DMPHN-WS(1-2)	29.27	0.9213	13
DMPHN-MPT(2-1)	29.10	0.9196	7
DMPHN-MST(1/2-1)	28.96	0.9184	7
DMPHN-MPT+MST(2-1-1/2)	29.19	0.9202	7
DMPHN-WS(1-2-4)	29.60	0.9261	22
DMPHN-MPT(4-2-1)	29.34	0.9223	7
DMPHN-MST(1/4-1/2-1)	29.22	0.9210	7
DMPHN-MPT+MST(4-2-1-1/2-1/4)	29.48	0.9246	7

The table 6.1 reports PSNR, SSIM and running time different model setups. The reported running time is CNN running time in milli-seconds. From the table it can be

seen that MPT+MST trained single stage network performs comparably with a multi-stage weight shared network. The speed up gained by training the network in a multi-stage fashion instead of using multi-stage network is proportional to the number of stages.

Having such small running time leaves room to make the network more complex and deep to further improve accuracy. It is also memory efficient and hence can be used on platforms with less computational resources.

6.2 Quantitative Results on Block-Wise Self-Attention for Image Deblurring

All our experiments are done on NVIDIA RTX 2080 TI GPU. We use the GoPro dataset [Nah *et al.* (2017)] for all our experiments. There are a total of 3214 blur and sharp image pairs of resolution 720x1280 in this dataset of which we use 2103 image pairs for training and remaining 1111 pairs for testing. We train all the networks on 256x256 patch randomly cropped from each image. We use batch size of 8 during training for all the results reported.

To have a fair comparison all the models reported below have approximately same number of parameters as the Aggregation Block is essentially equivalent (in terms of parameters) to two convolutional layers. The current experiments are performed with Deep Stacked Multi-patch Hierarchical Network (DMPHN) architecture as back-bone.

Table 6.2: Quantitative results for our model on GoPro dataset

Metric	DMPHN	DMPHN-AB(1,3)	DMPHN-AB(1,5)	DMPHN-AB(1,7)	DMPHN-AB(1,9)
PSNR	28.76	28.91	29.04	29.12	29.13
SSIM	0.9134	0.9152	0.9176	0.9191	0.9196

Metric	DMPHN-AB(1,15)	DMPHN-AB(3,9)	DMPHN-AB(3,15)	DMPHN-AB(5,15)
PSNR	29.15	29.31	29.54	29.46
SSIM	0.9197	0.9217	0.9248	0.9242

The table 6.2 reports PSNR, SSIM for Aggregation Blocks with multiple block sizes and kernel sizes. AB(x,y) represents a block size of x and kernel size of y. Block-size of one implies pixel-wise self-attention, as we can see the performance for block size

one increases with kernel size till the kernel size reaches 7 after which it saturates. At a kernel size of seven itself it is accumulating information from 49 units and as we are using softmax it can only accumulate information from few number of units, hence the performance saturates at kernel size of seven.

We can also see that block size of 3 gives better performance compared to block size of 1 for the same kernel size (9) which is in favor of our argument that using block-wise self attention is better. However, block size of 5 gives worse performance compared to block size of 3 for the same kernel size (15). We are adding the Aggregation Block after the encoder hence each position in the feature map has already accumulated information, using large block size such as 5 is probably more restrictive in terms of context aggregation.

6.3 Quantitative Results on Attentive Filtering for Visual Grounding

In this section we show our experiments on comparison with other state-of-the-art visual grounding methods. We perform all our experiments on RefCOCO dataset [Yu *et al.* (2016)] using the split as in [Yu *et al.* (2016)]. Which is train/val/testA/testB 120,624/10,834/5657/5095, testA contains images with multiple people and testB contains images of all the other objects. We also show some bounding box predictions on the RefCOCO dataset [Yu *et al.* (2016)].

All our experiments are done on NVIDIA RTX 2080 TI GPU. For fair comparison we use the same training procedure as Yang *et al.* (2019). We resize each image such that long size is of length 256 and pad pixels with mean value to the short edge to get an image of size 256x256. We perform data augmentation by adding randomization to color space, randomly flipping the images horizontally and random affine transformations. We use RMSProp [Tieleman and Hinton (2012)] as the optimizer to train the model. We use a batch size of 16 to train all the models.

6.3 shows comparison of our methods with other state-of-the-art methods OSVG [Yang *et al.* (2019)], Mattnet [Yu *et al.* (2018)], Similarity Net [Wang *et al.* (2019)], SLR [Yu *et al.* (2017)] and VC [Zhang *et al.* (2018)] using proposal methods SSD [Liu

Table 6.3: Accuracy on RefCOCO dataset[Yu *et al.* (2016)]. LSTM and COCO trained Res101 are encoders if not stated.

Model	Region Proposals	Val	testA	testB
SLR	FRCN	69.48	73.71	64.96
VC-VGG16	SSD	-	73.33	67.44
MattNet Base	FRCN	72.72	76.17	68.18
Mattnet	FRCN	**76.40**	**80.43**	69.28
Similarity Net	Edge box N=200	57.33	57.22	55.60
Similarity Net	FRCN	71.48	74.90	67.32
Similarity Net-Darknet	FRCN	72.27	75.12	67.91
OSVG-LSTM	None	72.05	74.67	70.12
Ours	None	74.02	76.43	**71.64**

et al. (2016)] or FRCN [Ren *et al.* (2017)] or Edge box [Zitnick and Dollár (2014)]. It is to be noted that we re-train only OSVG on our system, for other networks we take results from their respective papers. Our method produces 1.5% improvement over the previous one stage approach, hence we argue that directly fusing language and visual features is sub-optimal.

6.4 Qualitative Results on Attentive Filtering for Visual Grounding

In this section we see some results where our network performs better than OSVG [Yang *et al.* (2019)] and reason about them. The green box is the ground truth bounding box, blue box is the predicted bounding box and the language query is given as caption. In all the images below bounding box in the left image is predicted by OSVG and bounding box in the right image is predicted by our network.

Figure 6.1: falling

It can been that in all the example images [6.1, 6.4, 6.3, 6.5, 6.6, 6.2] that OSVG [Yang *et al.* (2019)] gives fairly accurate bounding boxes but for the wrong person which means that the information in the language query is not harvested fully. In our work we score portions of image that are being addressed in the language query and hence our network is better at recognizing the correct person.

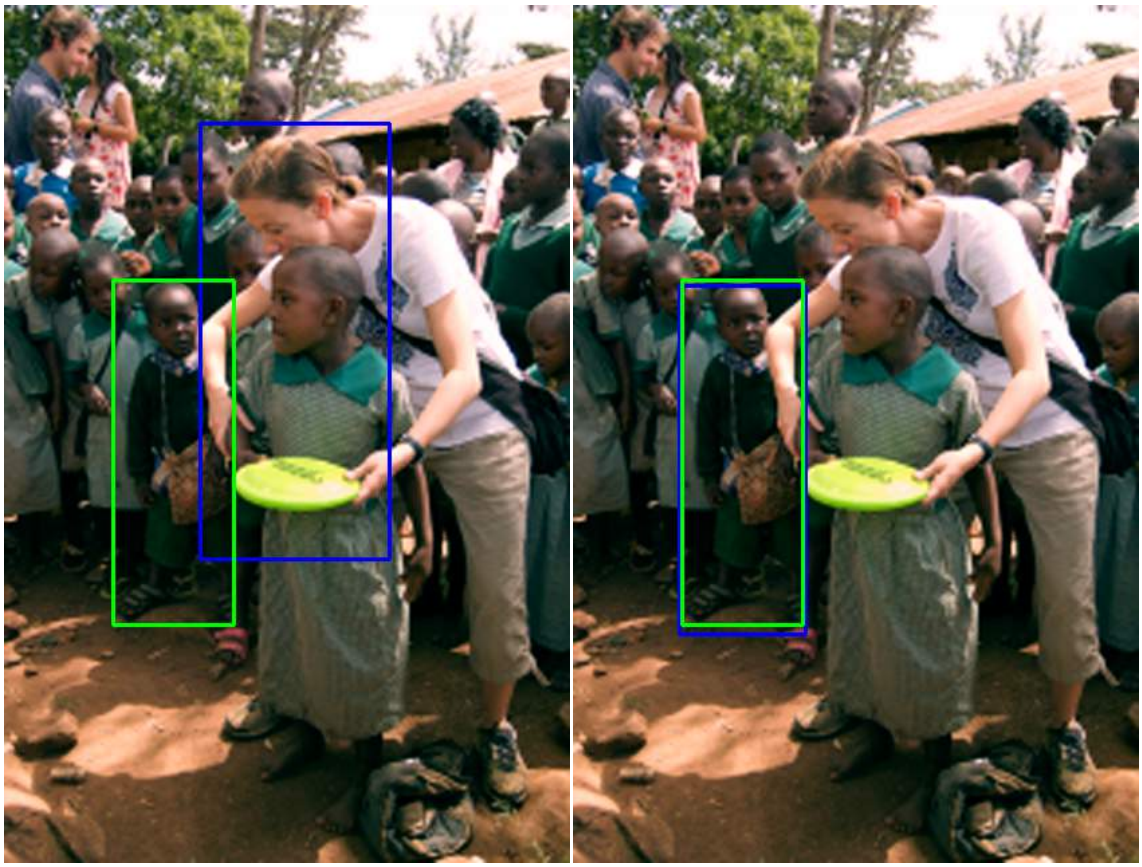

Figure 6.2: little boy left on womans elbow

Figure 6.3: granny

Figure 6.4: hands on coffee mug

Figure 6.5: guy in purple

Figure 6.6: girl with hands on head

CHAPTER 7

Conclusion

In this thesis we explore novel training procedures to improve performance of Single Image Deblurring networks which comes without any extra cost in terms of training time and inference time. We demonstrate the improvement in performance through experiments. These training procedures when applied on single stage networks give performance comparable to multi-stage networks all the while maintaining same inference time as single stage network. This makes the models very fast and memory efficient a perfect fit for platforms with less computational resources. We also review the concept of effective receptive field in the context of single image deblurring and propose novel block-wise self-attention block to aggregate the global context information, in effect greatly increasing the receptive field, resulting in better performance.

We also introduce a new feature filtering method for the task of Visual Grounding where we attentively filter out visual features that are not being referred to in the language query. This gives better performance by actively suppressing false positives compared to prediction of bounding boxes by appending language features to visual features. The feature filtering method we introduce can further be improved by using better attention mechanisms, hence we advocate the use and exploration of feature filtering method for the task of Visual Grounding.

REFERENCES

1. **Bello, I.**, **B. Zoph**, **A. Vaswani**, **J. Shlens**, and **Q. V. Le**, Attention augmented convolutional networks. *In The IEEE International Conference on Computer Vision (ICCV)*. 2019.

2. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun**, Deep residual learning for image recognition. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

3. **Hu, H.**, **Z. Zhang**, **Z. Xie**, and **S. Lin**, Local relation networks for image recognition. *In The IEEE International Conference on Computer Vision (ICCV)*. 2019.

4. **Hu, J.**, **L. Shen**, and **G. Sun**, Squeeze-and-excitation networks. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

5. **Hu, R.**, **H. Xu**, **M. Rohrbach**, **J. Feng**, **K. Saenko**, and **T. Darrell** (2016). Natural language object retrieval. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

6. **Ioffe, S.** and **C. Szegedy**, Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In* **F. Bach** and **D. Blei** (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*. PMLR, Lille, France, 2015. URL `http://proceedings.mlr.press/v37/ioffe15.html`.

7. **Krizhevsky, A.**, **I. Sutskever**, and **G. E. Hinton**, Imagenet classification with deep convolutional neural networks. *In Advances in neural information processing systems*. 2012.

8. **Lin, T.-Y.**, **P. Dollar**, **R. Girshick**, **K. He**, **B. Hariharan**, and **S. Belongie**, Feature pyramid networks for object detection. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

9. **Lin, T.-Y.**, **M. Maire**, **S. Belongie**, **J. Hays**, **P. Perona**, **D. Ramanan**, **P. Dollár**, and **C. L. Zitnick**, Microsoft coco: Common objects in context. *In* **D. Fleet**, **T. Pajdla**, **B. Schiele**, and **T. Tuytelaars** (eds.), *Computer Vision – ECCV 2014*. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10602-1.

10. **Liu, W.**, **D. Anguelov**, **D. Erhan**, **C. Szegedy**, **S. Reed**, **C.-Y. Fu**, and **A. C. Berg**, SSD: Single shot multibox detector. *In ECCV*. 2016.

11. **Luo, W.**, **Y. Li**, **R. Urtasun**, and **R. Zemel**, Understanding the effective receptive field in deep convolutional neural networks. *In* **D. D. Lee**, **M. Sugiyama**, **U. V. Luxburg**, **I. Guyon**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, 4898–4906. URL `http://papers.nips.cc/paper/6203-understanding-the-effective-receptive-field-in-deep-convolution` `pdf`.

12. **Mao, J.**, **J. Huang**, **A. Toshev**, **O. Camburu**, **A. Yuille**, and **K. Murphy**, Generation and comprehension of unambiguous object descriptions. *In CVPR*. 2016*a*.

13. **Mao, X.**, **C. Shen**, and **Y.-B. Yang**, Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. *In* **D. D. Lee**, **M. Sugiyama**, **U. V. Luxburg**, **I. Guyon**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016*b*, 2802–2810. URL http://papers.nips.cc/paper/6172-image-restoration-using-very-deep-convolutional-encoder-decoder pdf.

14. **Nah, S.**, **T. Hyun Kim**, and **K. Mu Lee**, Deep multi-scale convolutional neural network for dynamic scene deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

15. **Parmar, N.**, **P. Ramachandran**, **A. Vaswani**, **I. Bello**, **A. Levskaya**, and **J. Shlens**, Stand-alone self-attention in vision models. *In* **H. Wallach**, **H. Larochelle**, **A. Beygelzimer**, **F. d'Alché-Buc**, **E. Fox**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, 68–80. URL http://papers.nips.cc/paper/8302-stand-alone-self-attention-in-vision-models.pdf.

16. **Redmon, J.**, **S. Divvala**, **R. Girshick**, and **A. Farhadi**, You only look once: Unified, real-time object detection. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

17. **Redmon, J.** and **A. Farhadi** (2016). Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*.

18. **Redmon, J.** and **A. Farhadi** (2018). Yolov3: An incremental improvement. *arXiv*.

19. **Ren, S.**, **K. He**, **R. Girshick**, and **J. Sun** (2017). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, **39**(6), 1137–1149. ISSN 0162-8828. URL https://doi.org/10.1109/TPAMI.2016.2577031.

20. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-net: Convolutional networks for biomedical image segmentation. *In International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

21. **Simonyan, K.** and **A. Zisserman** (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

22. **Szegedy, C.**, **W. Liu**, **Y. Jia**, **P. Sermanet**, **S. Reed**, **D. Anguelov**, **D. Erhan**, **V. Vanhoucke**, and **A. Rabinovich**, Going deeper with convolutions. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

23. **Tao, X.**, **H. Gao**, **X. Shen**, **J. Wang**, and **J. Jia**, Scale-recurrent network for deep image deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

24. **Tieleman, T.** and **G. Hinton** (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

25. **Vaswani, A.**, **N. Shazeer**, **N. Parmar**, **J. Uszkoreit**, **L. Jones**, **A. N. Gomez**, **L. u. Kaiser**, and **I. Polosukhin**, Attention is all you need. *In* **I. Guyon**, **U. V. Luxburg**, **S. Bengio**, **H. Wallach**, **R. Fergus**, **S. Vishwanathan**, and **R. Garnett** (eds.), *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, 5998–6008. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

26. **Wang, L.**, **Y. Li**, **J. Huang**, and **S. Lazebnik** (2019). Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **41**, 394–407.

27. **Wang, X.**, **R. Girshick**, **A. Gupta**, and **K. He**, Non-local neural networks. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

28. **Yang, Z.**, **B. Gong**, **L. Wang**, **W. Huang**, **D. Yu**, and **J. Luo**, A fast and accurate one-stage approach to visual grounding. *In ICCV*. 2019.

29. **Yu, L.**, **Z. Lin**, **X. Shen**, **J. Yang**, **X. Lu**, **M. Bansal**, and **T. L. Berg**, Mattnet: Modular attention network for referring expression comprehension. *In CVPR*. 2018.

30. **Yu, L.**, **P. Poirson**, **S. Yang**, **A. C. Berg**, and **T. L. Berg**, Modeling context in referring expressions. *In ECCV*. 2016.

31. **Yu, L.**, **H. Tan**, **M. Bansal**, and **L. T. Berg** (2017). A joint speaker-listener-reinforcer model for referring expressions. *CVPR*.

32. **Zhang, H.**, **Y. Dai**, **H. Li**, and **P. Koniusz**, Deep stacked hierarchical multi-patch network for image deblurring. *In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

33. **Zhang, H.**, **Y. Niu**, and **S. Chang**, Grounding referring expressions in images by variational context. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

34. **Zitnick, C. L.** and **P. Dollár**, Edge boxes: Locating object proposals from edges. *In* **D. Fleet**, **T. Pajdla**, **B. Schiele**, and **T. Tuytelaars** (eds.), *Computer Vision – ECCV 2014*. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10602-1.