This dissertation, directed and approved by Aaditya Prakash's committee, has been accepted and approved by the Graduate Faculty of Brandeis University in partial fulfillment of the requirements for the degree of:

DOCTOR OF PHILOSOPHY

Eric Chasalow, Dean of Arts and Sciences

Dissertation Committee:

James A. Storer, Chair

Antonella DiLillo

Sadid Hasan

Thesis

A Dissertation

Presented to

The Faculty of the Graduate School of Arts and Sciences

Brandeis University

Michtom School of Computer Science

James A. Storer, Advisor

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

Aaditya Prakash

May, 2019

This dissertation, directed and approved by Aaditya Prakash's committee, has been accepted and approved by the Graduate Faculty of Brandeis University in partial fulfillment of the requirements for the degree of:

DOCTOR OF PHILOSOPHY

Eric Chasalow, Dean of Arts and Sciences

Dissertation Committee:

James A. Storer, Chair

Antonella DiLillo

Sadid Hasan

Acknowledgments

I would like to thank my advisor James Storer for the wonderful time I have had during my graduate school years. He provided a perfect amount of supervision while letting me explore various areas of research that interested me. He made sure that I had a good work-life balance and that I was happy and passionate about the research. I could not have asked for a better advisor and a mentor.

I would like to thank Sadid Hasan (Philips Research), Raghuraman Krishnamoorthi (Qualcomm), Dinei Florencio (Microsoft) and Cha Zhang (Microsoft) who were my mentors during my summer internships and from whom I got to learn tremendously.

I am indebted to my friend Nick Moran who not only proofread all the papers that constitute this thesis but also served as a sounding board for all the ideas.

I would also like to thank Solomon Garber, Ryan Marcus and Antonella Di Lillo for various ideas and discussions throughtout my graduate years and for proof-reading all my papers.

I would like to acknowledge generous grants and donations by NVIDIA, Intel and Google, which made possible most of the research.

Last but not least I would like to thank my parents Meena Karna and Vinay K Karna for always encouraging me to strive for the best.

Abstract

Thesis

A dissertation presented to the Faculty of
the Graduate School of Arts and Sciences of
Brandeis University, Waltham, Massachusetts

by Aaditya Prakash

Preface

This thesis is comprised of various published research. Chapters are divided as independent papers and provides all necessary introductions and related literature. Here is a brief description of the chapters.

Chapter One provides a brief discussion of Neural Networks and Convolutional Neural Networks that is necessary to follow the subsequent chapters. Only necessary elaboration is provided and user is encouraged to explore various textbooks in the area for a complete and a thorough guide.

Chapter two presents the research which shows how to use Convolutional Neural Network to make image compression that is semantically aware. It has long been considered a significant problem to improve the visual quality of lossy image and video compression. Recent advances in computing power together with the availability of large training data sets has increased interest in the application of deep learning cnns to address image recognition and image processing tasks. Here, we present a powerful cnn tailored to the specific task of semantic image understanding to achieve higher visual quality in lossy compression. A modest increase in complexity is incorporated to the encoder which allows a standard, off-the-shelf jpeg decoder to be used. While jpeg encoding may be optimized for generic images, the process is ultimately unaware of the specific content of the image to be compressed. Our technique makes jpeg content-aware by designing and training a model to identify multiple semantic regions in a given image. Unlike object detection techniques, our model does not require labeling of object positions and is able to identify objects in a single pass. We present a new cnn architecture directed specifically to image compression, which generates a map that highlights semantically-salient regions so that they can be encoded at higher quality as compared to background regions. By adding a complete set of features for every class, and then taking a threshold over the sum of all feature activations, we generate a map that highlights semantically-salient regions so that they can be encoded at a better quality compared to background regions. Experiments are presented on the Kodak PhotoCD dataset and the MIT Saliency Benchmark dataset, in which our algorithm achieves higher visual quality for the same compressed size. This chapters presents the work published as –

Prakash, Aaditya, Nick Moran, Solomon Garber, Antonella DiLillo and James A. Storer. *"Semantic Perceptual Image Compression Using Deep Convolution Networks."* 2017 Data Compression Conference (DCC - Oral) (2017)

Chapter three presents the research which improves upon the results from chapter one by making these images robust in the presence of an adversary. As deep neural networks (DNNs) have been integrated into critical systems, several methods to attack these systems have been developed. These adversarial attacks make imperceptible modifications to an image that fool DNN classifiers. We present an adaptive JPEG encoder which defends against many of these attacks. Experimentally, we show that our method produces images with high visual quality while greatly reducing the potency of state-of- the-art attacks. Our algorithm requires only a modest increase in encoding time, produces a compressed image which can be decompressed by an off-the-shelf JPEG decoder, and classified by an unmodified classifier. This chapters presents the work published as –

Prakash, Aaditya, Nick Moran, Solomon Garber, Antonella DiLillo and James

PREFACE

A. Storer. *“Protecting JPEG Images Against Adversarial Attacks.”* 2018 Data Compression Conference (DCC - Oral) (2018)

Chapter four extends the work of preventing use of adversarial images from being used to fool the deep networks. CNNs are poised to become integral parts of many critical systems. Despite their robustness to natural variations, image pixel values can be manipulated, via small, carefully crafted, imperceptible perturbations, to cause a model to misclassify images. We present an algorithm to process an image so that classification accuracy is significantly preserved in the presence of such adversarial manipulations. Image classifiers tend to be robust to natural noise, and adversarial attacks tend to be agnostic to object location. These observations motivate our strategy, which leverages model robustness to defend against adversarial perturbations by forcing the image to match natural image statistics. Our algorithm locally corrupts the image by redistributing pixel values via a process we term pixel deflection. A subsequent wavelet-based denoising operation softens this corruption, as well as some of the adversarial changes. We demonstrate experimentally that the combination of these techniques enables the effective recovery of the true class, against a variety of robust attacks. Our results compare favorably with current state-of-the-art defenses, without requiring retraining or modifying the CNN. This chapters presents the work published as –

Prakash, Aaditya, Nick Moran, Solomon Garber, Antonella DiLillo and James A. Storer. *“Deflecting Adversarial Attacks with Pixel Deflection.”* 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR - Spotlight) (2018)

Chapter five finds yet another limitation of deep networks - redundancy in feature representation. A well-trained Convolutional Neural Network can easily be pruned without significant loss of performance. This is because of unnecessary overlap in the features captured by the network's filters. Innovations in network architecture such as skip/dense connections and Inception units have mitigated this problem to some extent, but these improvements come with increased computation and memory requirements at run-time. We attempt to address this problem from another angle - not by changing the network structure but by altering the training method. We show that by temporarily pruning and then restoring a subset of the model's filters, and repeating this process cyclically, overlap in the learned features is reduced, producing improved generalization. We show that the existing model-pruning criteria are not optimal for selecting filters to prune in this context and introduce inter-filter orthogonality as the ranking criteria to determine under-expressive filters. Our method is applicable both to vanilla convolutional networks and more complex modern architectures, and improves the performance across a variety of tasks, especially when applied to smaller networks. This chapters presents the work published as –

Prakash, Aaditya, James A. Storer, Dinei A. F. Florêncio and Cha Zhang. *"RePr: Improved Training of Convolutional Filters."* 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR - Oral) (2019)

Chapter six gives a brief description of how to extend Convolutional Neural Networks to incorporate languages when the task involves multimodal signals. We propose a version of highway network designed for the task of Visual Question Answering. We take inspiration from recent success of Residual Layer Network and Highway Network in learning deep representation of images and fine grained localization of objects. We propose variation in gating mechanism to allow incorporation of word embedding in the information highway. The gate parameters are influenced by the words in the question, which steers the network towards localized feature learning. This achieves the same effect as soft attention via recurrence but allows for faster training using optimized feed-forward techniques. We are able to obtain state-of-the-art1 results on VQA dataset for Open Ended and Multiple Choice tasks with current model. This chapters presents the work published as –

Prakash, Aaditya and James Storer Brandeis. *"Highway Networks for Visual Question Answering."* IEEE/CVF Conference on Computer Vision and Pattern Recognition (VQA - Workshop - Spotlight) (2016)

Chapter seven goes deeper into languge domains and explores the idea of residual connection in the context of languages. we propose a novel neural approach for paraphrase generation. Conventional paraphrase generation methods either leverage handwritten rules and thesauri-based alignments, or use statistical machine learning principles. To the best of our knowledge, this work is the first to explore deep learning models for paraphrase generation. Our primary contribution is a stacked residual LSTM network, where we add residual connections between LSTM layers. This allows for efficient training of deep LSTMs. We experiment with our model and other state-of-the-art deep learning models on three different datasets: PPDB, WikiAnswers and MSCOCO. Evaluation results demonstrate that our model outperforms sequence to sequence, attention-based and bi-directional LSTM models on BLEU, METEOR, TER and an embedding-based sentence similarity metric. This chapters presents the work published as –

Prakash, Aaditya, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu and Oladimeji Farri. *"Neural Paraphrase Generation with Stacked Residual LSTM Networks."* COLING (2016)

PREFACE

Chapter eight explores memory networks and shows its efficacy in diagnosis of diseses. Diagnosis of a clinical condition is a challenging task, which often requires significant medical investigation. Previous work related to diagnostic inferencing problems mostly consider multivariate observational data (e.g. physiological signals , lab tests etc.). In contrast, we explore the problem using free-text medical notes recorded in an electronic health record (EHR). Complex tasks like these can benefit from structured knowledge bases, but those are not scalable. We instead exploit raw text from Wikipedia as a knowledge source. Memory networks have been demonstrated to be effective in tasks which require comprehension of free-form text. They use the final iteration of the learned representation to predict probable classes. We introduce condensed memory neural networks (C-MemNNs), a novel model with iterative condensation of memory representations that preserves the hierarchy of features in the memory. Experiments on the MIMIC-III dataset show that the proposed model outperforms other variants of memory networks to predict the most probable diagnoses given a complex clinical scenario. This chapters presents the work published as –

Prakash, Aaditya, Siyuan Zhao, Sadid A. Hasan, Vivek Datla, Kathy Lee, Ashequl Qadir, Joey Liu and Oladimeji Farri. *"Condensed Memory Networks for Clinical Diagnostic Inferencing."* AAAI (2017)

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Introduction to Neural Networks

Neural Networks are a type of a machine learning model inspired by activities in human brain. A neural network comprises of ‘neurons‘, which are considered as the building blocks. A neuron in a neural network is implemented as weighted sum of its input, which is then passed through a non-linear function. This non-linear function is often denoted as an activation function and most commonly is a sigmoid or a rectified linear function.

Let W be weights and x denote the inputs. Let g be the activation function, then a single neuron is represented as -

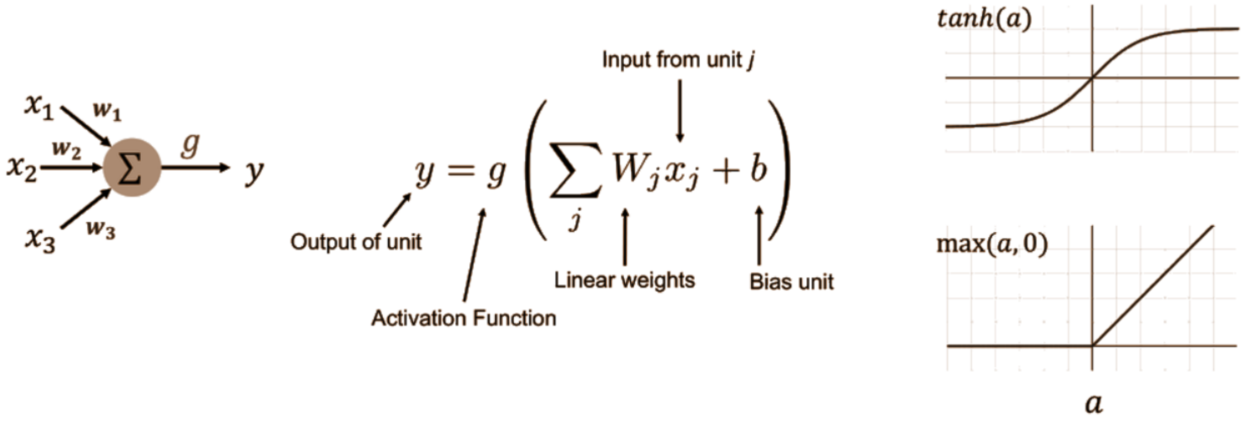

Figure 1.1: Artificial Neuron and common activation functions

When multiple of neurons are connected in a layer to all the inputs and the output of these neurons are used as input to another set of neurons, then the structure is called as Neural Network. This is shown in the figure 1.2. In this network x_1 and x_2 are the inputs to the network. $f_1(e)$, $f_2(e)$ and $f_3(e)$ are the outputs of three neurons in the first layer. They form as the inputs to the second layer. Simillarly $f_4(e)$ and $f_5(e)$ are the outputs of second layer and $f_6(e)$ is the final output which is used to predict the final value - y.

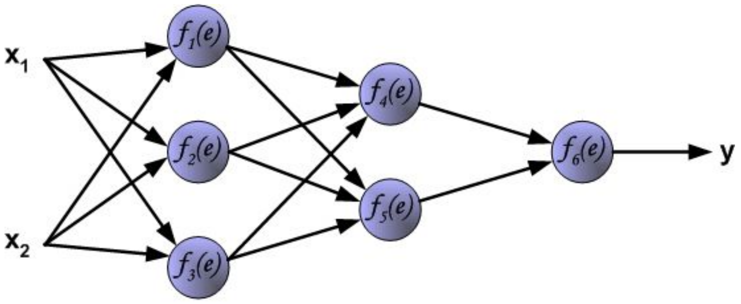

Figure 1.2: Artificial Neuron and common activation functions

1.1.1 Learning

Learning of Neural Networks is done in four steps –

1. Forward pass

   During the forward pass the outputs are computed by multiplying the weights w with the inputs (x). Let $w \times x$ be denoted as e and the activation function be denoted as f, then output of a layer $f(e)$ would be $f_1(w_1 \times x_1 + w_2 \times x_2)$. If we consider the network shown in figure 1.2, then a single forward pass would be as depicted in the figure 1.3.

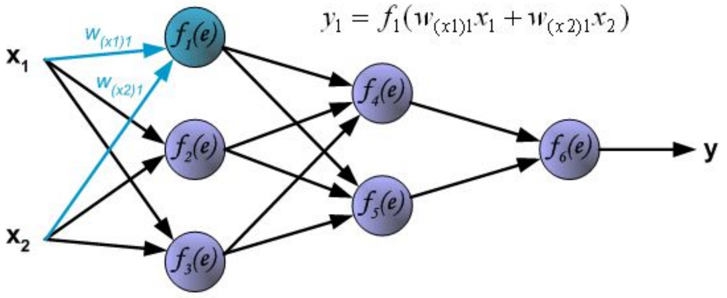

Figure 1.3: Computation of a forward pass in a neural network

2. Compute error

   Error is computed once the model has predicted its output (y). Let z be the true output for the given data points. Error function or the loss function takes in the predicted value (y) and the true value (z) and returns some numeric value. This function is task specific and is different for different tasks. For simplicity let's assume a simple difference, then the error δ is $z - y$, as shown in the figure 1.4.

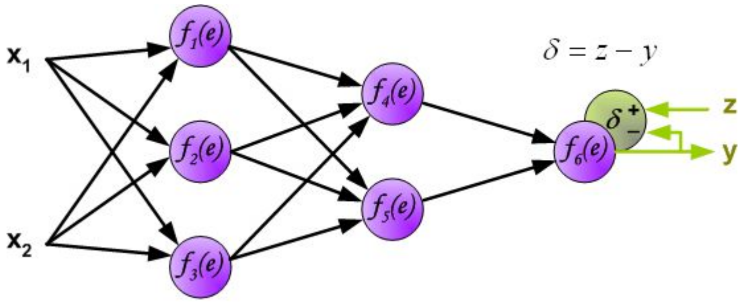

Figure 1.4: Computation of error in a neural network

3. Backward pass

Backward pass is a phase where error with respect to every neuron is computed. Error at a given layer is the proportion of the error contributed by that layer wrt to the total error. In order to compute error for layer l we need to know error at layer $l+1$. Thus, it is computed from layer layer to the first layer and hence called Backward pass. For the networks shown above, error at neuron 2 is weighted sum of errors at neuron 4 and 5. This is shown in the figure 1.5.

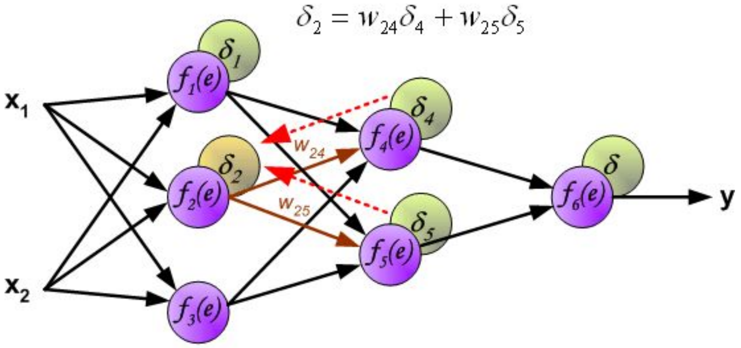

Figure 1.5: Backward pass of the error in a neural network

4. Weight update

Final step is to use the error at each neuron to change the weights. Current weight is changed by product of three values - error at the current neuron (δ), gradient of the output wrt to the input at that node ($\frac{df(e)}{de}$)and the output y. Generally, this update value is too large and needs to be scaled. This scaling parameter (η) is called as the **learning rate** and is generally a hyper-parameter of the network. This step is shown in the figure 1.6.

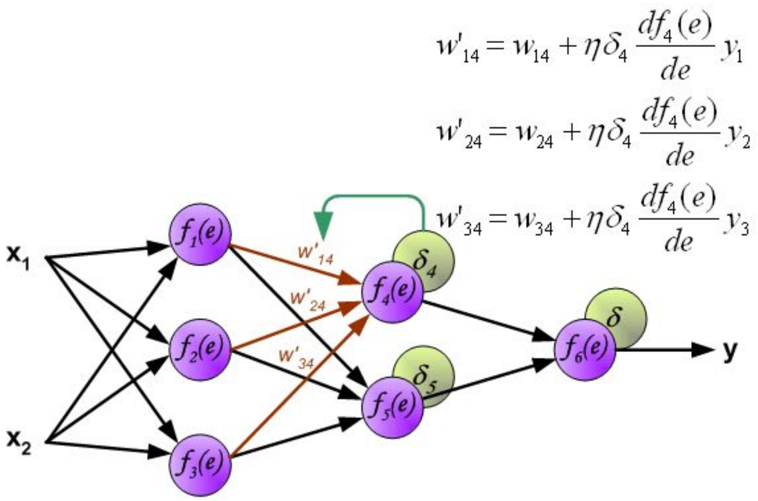
$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$
$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$
$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$
x_1
x_2
$f_1(e)$
$f_2(e)$
$f_3(e)$
w'_{14}
w'_{24}
w'_{34}
δ_4
$f_4(e)$
δ_5
$f_5(e)$
δ
$f_6(e)$
y

Figure 1.6: Update of weights in a neural network

1.2 Introduction to Convolutional Neural Networks

Convolutional Neural Network (CNN) are like standard neural networks except that not all neurons are connected to every input. Only local neighbourhood of inputs are shared for some set of neurons as depicted. Figure 1.7 shows the difference between standard Neural Networks (left) and Convolutional Neural Neworks (right).

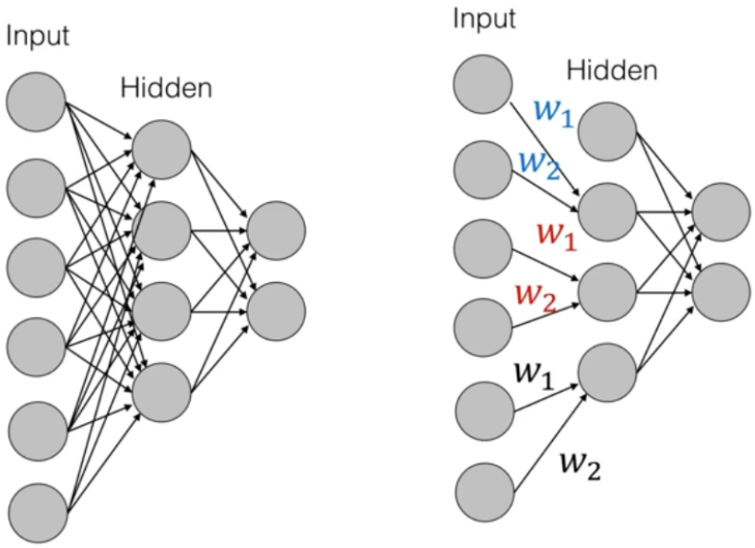

Figure 1.7: Difference between Standard Neural Network (left) and Convolutional Neural Network (Right)

1.2.1 Convolution

While figure 1.7 shows CNN in one-dimensional input, it is most widely used with images which are 2-D. For 2-D data the local inputs shared are a smaller 2-D window in the given image. In order to compute the output a smaller 2-D weights (filter) is convolved across the image. Generally, this leads to 2-D output but with slighly smaller spatial dimensions due to lack of enough values during convolving at the extreme points. This kind of convolution is called as 'VALID' convolution. In order to achieve the same spatial output the input can be padded wit zeros. This kind of convolution is commonly referred to as 'SAME' convolution. An example of VALID 2-D convolution is shown in the figure 1.8. Resulting outputs are called as activation maps. If a layer learns k filters, then there are k activation maps as the output.

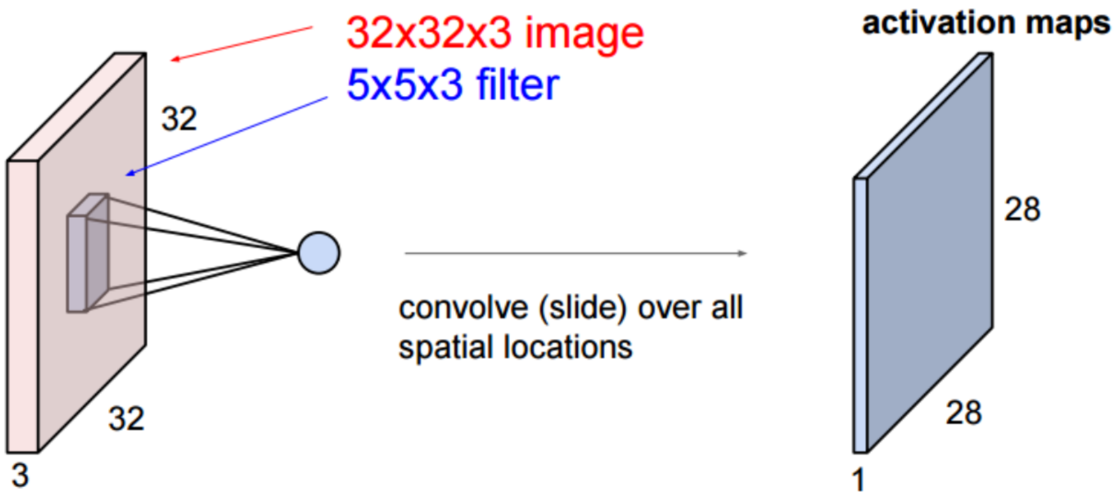

Figure 1.8: VALID 2-D Convolution

1.2.2 Pooling

Convolutional Neural Networks take in images of some size $H \times W$ and generally, output fixed scalar values in $\mathbb{R}^n$, where $n << H, W$. This, often necessitates decreasing the dimensions of the activation maps signficantly (much more than the loss of edges due to VALID convolution). One common technique used to decrease the spatial dimensions is to pool the values within a smaller window. For all values in the window of $h \times w$ is replaced by a single value. Quite often the aggregate function used is the **max** operation however some models also use averag of the values. Pooling operation is depicted in the figure **??**.

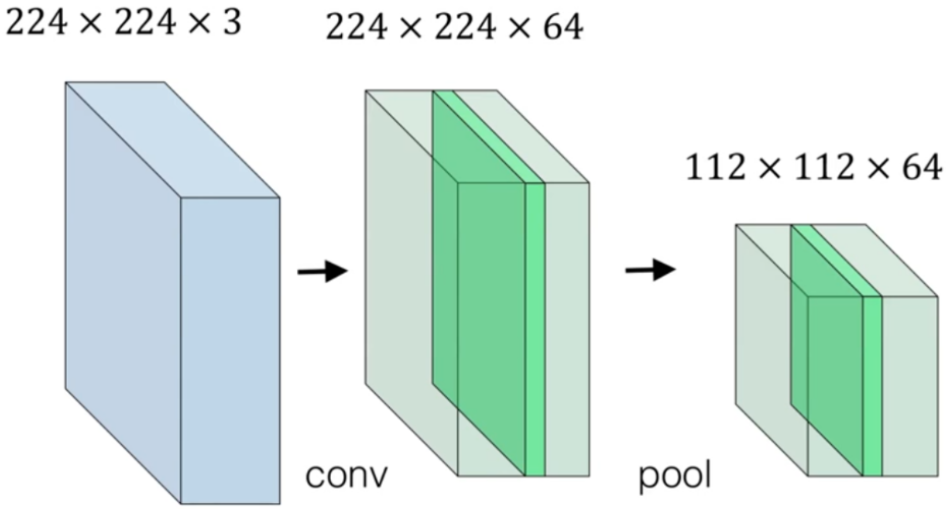

Figure 1.9: Pooling

Chapter 2

Multi-structure Regions of Interest

Chapter 3

Robust regions of interest

Chapter 4

Pixel Deflection

Chapter 5

RePr

Chapter 6

Multimodal Highway Networks

Chapter 7

Residual LSTM

Chapter 8

Condensed Memory Networks