UNIVERSITY OF CALIFORNIA, MERCED

Biologically Inspired Efficiencies in Computer Vision and Audition

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Mohammad K. Ebrahimpour

Committee in charge:

Professor David C. Noelle, Chair
Professor Ming-Hsuan Yang
Professor Shawn Newsam
Professor Christopher T. Kello

2020

The dissertation of Mohammad K. Ebrahimpour is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Professor Ming-Hsuan Yang

Professor Shawn Newsam

Professor Christopher T. Kello

Professor David C. Noelle Chair

University of California, Merced

2020

DEDICATION

To my wife Parisa,
who encourages and inspires me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My path to earning a PhD degree has been long. It started back in the kindergarten, when I was curious about everything. From primary to high school I was taught how to read, write, add/subtract, and identify right and wrong. It was in my undergraduate program that I learned how to fall seven times and stand up eight times. I experienced the joy of a novel discovery and publishing pear-reviewed papers when I was in my first grad-school, while I was doing my Masters degree. After that, I started the wonderful journey of the PhD. Now that I am at the finish line in the PhD program, I know that the term "philosophy" in PhD might have another meaning: *love of wisdom*. It was this love of wisdom which encouraged me to start this journey and to stay intellectually hungry, and I hope that it will never leave me.

As I said, the path to becoming a doctor was long, but it was full of amazing people to acknowledge.

First, I would like to thank Dr. David Noelle, my PhD advisor. When I first knocked on his door, he treated me with great respect, and he introduced me to a clear research path. He always had time for me whenever I needed to have a discussion with him despite his very busy schedule. During the PhD years, he framed my thinking style, and I'll always be grateful for this. During my time in his lab, I tried to learn everything from him, from the way that he writes his emails to the way that he sees the world. After all of these years, I learned how to be polite and humble while being a very respected scientist from him. He was very supportive and knowledgeable, and I consider myself very lucky that I had this opportunity to work under his supervision during my PhD years. Thanks David for everything!

Most of all, I would like to sincerely thank my PhD committee members:

Dr. Chris Kello for all of his support. He gave me the courage to explore new ideas in a new field. He also gave me brilliant advice for determining my future career path. I enjoyed every moment of our collaboration during the past year.

Dr. Ming-Hsuan Yang for his great advice toward visual understanding part of dissertation and also for sharing his great notes on scientific writing. He also generously allowed me to use hardware from his lab.

Dr. Shawn Newsam for his great advice and his great personality. I also really enjoyed

his course on “Digital Image Processing”.

I would like to thank my lab mates Jeff, Tim, Jacob , and Narjes. Not only did they help me with everything I needed, but we also became good friends. The coffee break was the best moment of the working day. Not only for the coffee, nor for the break, but for the nice company. Thanks for that.

My most sincere gratitude goes toward Ramin Raziperchicholaei for great discussions. For almost 3 years we had tea break along with critical discussions every afternoon at 4 PM.

In this journey, I also had the opportunity to enrich my industrial experience by squeezing in internships as a computer vision research scientist and a resident research scientist at Ancestry Inc. and Accenture Labs. I learned a lot during my stay with both of them.

I am truly thankful to all my friends for spending our leisure time together, bringing nothing but good moments to my life. A special mention goes to Ramin Raziperchicholaei and Maryam Shadloo. Your friendship is a gift to me, and I will always treasure it.

There aren’t enough words to express how grateful I am to my family. My parents, Habib and Fatemeh, are my heroes. They made my education the first priority and kept me on track until now. My brother, Sadegh, supported me in many difficult situations and I knew I always could count on him. *There is no better friend than a brother. And there is no better brother than you.*

Last but not least, I want to thank my wife Parisa. She was with me shoulder to shoulder in difficult times and she has always supported me in this journey. She was very patient with me, especially on final days and nights prior to paper submission deadlines. I cannot think of a better person to share my life with.

VITA

2015	M. Sc. in Artificial Intelligence, Shahid Bahonar University of Kerman, Iran
2020	Ph. D. in Electrical Engineering and Computer Science, University of California, Merced

PUBLICATIONS

M.K. Ebrahimpour, D.C. Noelle, *Weakly Supervised Object Localization via Sensitivity Analysis*, In Deep Vision Workshop at Computer Vision and Pattern Recognition (CVPR), 2018.

M.K. Ebrahimpour, J.Li, M.H,Yang, Y.Y.Yu, J.Reese, A,Moghtaderi, D.C. Noelle, *Ventral-Dorsal Networks: Object Detection via Selective Attention*, In IEEE Winter Conf. on Applications of Computer Vision (WACV), 2019.

M.K. Ebrahimpour, D.C. Noelle, *On the Interpretability of Deep Neural Networks: Object Localization with Sensitivity Analysis*, In International Symposium on Visual Computing (ISVC), 2019.

M.K. Ebrahimpour, B.Falandays, S.Spevack,D.C. Noelle,*Do Humans Look Where Deep Convolutional Neural Networks "Attend"?*, In Cognitive Science Society (CogSci), 2019.

M.K. Ebrahimpour, B.Falandays, S.Spevack,D.C. Noelle,*Do Humans Look Where Deep Convolutional Neural Networks "Attend"?*, In International Symposium on Visual Computing (ISVC), 2019.

M.K. Ebrahimpour, B.Falandays, S.Spevack, M.H.Yang, D.C. Noelle, *Dual Neural Networks for Object Detection*, In International Joint Conference on Neural Networks (IJCNN), 2020.

M.K. Ebrahimpour, T.M.Shea, A.Danielescu, D.C.Noelle, C.Kello, End-to-End Auditory Object Recognition via Inception Nucleus, In International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2020.

M.K. Ebrahimpour, T.M.Shea, A.Danielescu, D.C.Noelle, C.Kello, End-to-End Auditory Object Recognition on Neuromorpic hardware chip, In TinyML 2020.

M.K. Ebrahimpour, S.Schneider, D.C.Noelle, C.Kello, InfantNet: A Deep Neural Network for Analyzing Infant Vocalizations, In submission of Interspeech 2020.

ABSTRACT OF THE DISSERTATION

Biologically Inspired Efficiencies in Computer Vision and Audition

by

Mohammad K. Ebrahimpour
Doctor of Philosophy in Electrical Engineering and Computer Science
University of California Merced, 2020
Professor David C. Noelle, Chair

Computer perception is one of the fundamental problems in artificial intelligence. Given an image or an recorded audio, a human can quickly recognize and detect objects based on image or sound or both. In computer vision, Object Detection is concerned with recognizing objects in images and drawing a bounding box around them. Researchers have been working on developing algorithms to recognize, detect , and segment objects/scenes in images for decades. Numerous challenges make these problems significantly challenging in real world scenarios, since objects usually appear in different conditions, such as viewpoints, scales, and with background noise, and they even may deform into different shapes, parts, or poses. Real time object detection has many important applications, such as autonomous driving cars and video surveillance. In this dissertation, we approach visual understanding in the following ways: First, we utilize implicit information in trained neural networks to localize all objects of interest in an image using a sensitivity analysis approach. Second, we introduce a novel framework for object detection called "Ventral-Dorsal" Neural Networks, inspired by the structure of human brain. Third, we expand the Ventral-Dorsal framework, focusing on attaining real time performance needed for online applications. Forth, we compare human attention with deep neural network attention algorithms in order to understand whether neural network attention matches human attention. Also, auditory perception is crucial in artificial intelligence systems. Until recently, auditory object recognition pipelines were in need of substantial hand engineering for feature extraction. Engineered features need to be tuned for every individual problem. Also, some popular feature extraction methods are time consuming, limiting real time applications.

Here we attempt to avoid these problems using end-to-end training. Due to the recent improvements in deep neural networks, we are able to eliminate feature learning by optimizing feature extraction and classification jointly in one network. In this dissertation, we approach the auditory object recognition problem in the following ways: we proposed a novel "end-to-end" deep neural network architecture that takes raw audio as input and maps it to class labels. We also applied our proposed architecture to a new dataset of infant vocalization sounds for further investigation.

Chapter 1

Introduction

1.1 Overview

Computer perception is one of the fundamental problems in artificial intelligence. Given an image or an audio stream, a human can quickly recognize and detect objects based on image or sound or both. The research in this dissertation involves computer perception using visual and audio inputs. We have developed some object detection and visual understanding techniques, as well as end-to-end auditory object recognition algorithms. The topics that we worked on are as follows:

1) Does a deep neural network which can recognize different classes of objects contain any information about the location of objects [24]?

2) Why are humans good at object detection? What does this suggest to computer vision [23]?

3) Can we create a computer vision framework that runs in realtime while employing a sophisticated attentional mechanism [22]?

4) Do deep convolutional neural networks (CNNs) and people use visual attention in the same way [21]?

5) Since deep neural networks are capable of eliminating the feature extraction step in image recognition can we have an end-to-end auditory object recognition network that gets raw audio inputs and maps them to labels without extracting hand engineered features [20]?

6) Given a dataset of infant voices containing infant vocalizations, non-vocalizations, infant directed speech, adult directed speech and cry/laugh voices, can deep neural network learn

representations in order to classify these categories of sounds effectively [19]?
This dissertation can be divided into two main parts. The first part discusses computer perception of images and the second part investigates computer perception of auditory streams.

Computer Perception, Involving Visual Understanding

Visual understanding is one of the important parts of artificial intelligence. Given an image, a person can quickly identify the objects in the image and can detect all objects of interest.

In real world problems, detecting all objects of interest is a hard task for computers due to the variation of objects with respect to size, shape, and aspect ratios. This problem can get even harder when we are dealing with objects in perspective.

Another challenging factor of this problem is associated with the applications. In most object detection applications the results should be fast to avoid catastrophic consequences. For instance, if we have an autonomous driving car, since the environment is dynamic, the decision making process from the visual understanding side of the pipeline must be in real time since the final decision is a life and death decision.

Computer Perception Involving Audio Analysis

Another important perceptual ability involves audio. Audio analysis is important in a variety of ways. For example, sometimes in video surveillance the lighting situation is not very good, so the image or video is not meaningful. In this scenario, having audio perception would come in handy. For instance, if a surveillance camera records events at night with bad lighting, analyzing the sounds makes more sense when trying to determine the events.

There are times when the only relevant stimulus is sound. For instance, during the Covid-19 pandemic, a useful tool might be, one that detects infection by analyzing the sound of one's coughing sound.

1.2 Solutions

Visual Analysis

We have introduced the problem of visual analysis and its challenges. Recently, a number of deep learning frameworks [32, 58, 35] have shown significant improvement on object recognition tasks. However, these methods generally propose the use of one neural network for solving an object detection task. Since the performance of these models is still far from human performance levels, we hypothesized that biological vision could inform the

improvement of the models. In this dissertation, we are planning to introduce two concepts to object detection frameworks. One of the reasons that human beings are performing well on the object detection task is because they have *Selective Attention*. This means our brain evolved to pay attention to the gist of the visual field and ignore the irrelevant information. Second, in our studies, we noticed that even our brain does not solve the detection problem with one network. In our brain we have two pathways for "What" and "Where" determinations. Given a brief introduction, we would like to answer the following two questions:

- How can we incorporate selective attention into CNNs?
- How can we implement attention using two pathways for object detection?

In response to the first question, we proposed a number of gradient based attentional models to resemble selective attention. In response to the second question, we proposed a novel framework called ventral-dorsal neural networks for object detection, which resembles the ventral and dorsal pathways in the human brain.

Audio Analysis

The majority of auditory object recognition pipelines extract hand engineered features, like Mel-Frequency Cepstrul Coefficients (MFCC) [52], and they treat the extracted features as an image, allowing for the application of a CNN on it for the recognition task. But one of the advantages of using CNNs is that they are able to extract informative features by themselves. Given this fact, we investigated some potential architectures for end-to-end auditory object recognition. The main questions that we want to answer are as follows:

- Can we input time domain audio signal to a neural network and map it to class labels?
- What kind of representations will the neural network learn with such an end-to-end approach?

To answer the above questions, we proposed a novel architecture that takes a raw time domain signal and maps it to labels. We also investigated the learned representations to shed light on the working of our proposed model.

1.3 Organization

In Chapter 2, we offer a novel approach to the localization of recognized objects in images. Our method is predicated on the idea that a deep CNN capable of recognizing an object must implicitly contain knowledge about object location in its connection weights. We provide a simple method to interpret classifier weights in the context of individual classified images. This method involves the calculation of the derivative of network generated activation patterns, such as the activation of output class label units, with regard to each input pixel, performing a sensitivity analysis that identifies the pixels that, in a local sense, have the greatest influence on internal representations and object recognition. These derivatives can be efficiently computed using a single backward pass through the deep CNN classifier, producing a *sensitivity map* of the image. We demonstrate that a simple linear mapping can be learned from sensitivity maps to bounding box coordinates, localizing the recognized object. Our experimental results, using real-world data sets for which ground truth localization information is known, reveal competitive accuracy from our fast technique.

In Chapter 3, we propose a new framework, called Ventral-Dorsal Networks (VDNets), which is inspired by the structure of the human visual system. Roughly, the visual input signal is analyzed along two separate neural streams, one in the temporal lobe and the other in the parietal lobe. The coarse functional distinction between these streams is between object recognition — the "what" of the signal – and extracting location related information — the "where" of the signal. The ventral pathway from primary visual cortex, entering the temporal lobe, is dominated by "what" information, while the dorsal pathway, into the parietal lobe, is dominated by "where" information. Inspired by this structure, we propose the integration of a "Ventral Network" and a "Dorsal Network", which are complementary. Information about object identity can guide localization, and location information can guide attention to relevant image regions, improving object recognition. This new dual network framework sharpens the focus of object detection. Our experimental results reveal that the proposed method works favorably in comparison to the state of the art approaches.

In Chapter 4, we propose a new deep convolutional neural network framework that uses object location knowledge implicit in network connection weights to guide selective attention in object detection tasks. The VDNets of Chapter 3 are augmented to produce a

framework called Dense VDNets. The aim of the Ventral Network is to provide selective attention to the relevant parts of the input image. The Dorsal Network uses this information to locate and classify objects of interest. We compare this approach to state-of-the-art algorithms on the PASCAL VOC 2007 and 2012 and COCO object detection challenge datasets.

In Chapter 5, we investigate the reasons why Deep Convolutional Neural Networks (CNNs) exhibit relatively poor performance on object detection tasks. We hypothesize that this gap in performance may be largely due to the fact that humans exhibit *selective attention*, while most object detection CNNs have no corresponding mechanism. In examining this question, we investigate some well-known attention mechanisms in the deep learning literature, identifying their weaknesses and leading us to propose a novel CNN approach to object detection: the *Densely Connected Attention Model*. We then report the results of a study of human spatial attention, using eye tracking data, during the performance of an analogous object detection task. By comparing the learned representations produced by various CNN architectures with the fixations exhibited by human viewers, we identify some relative strengths and weaknesses of the examined computational attention mechanisms. Some CNNs produced attentional patterns somewhat similar to those of humans. Others focused processing on objects in the foreground. Still other CNN attentional mechanisms produced usefully interpretable internal representations. The resulting comparisons provide insights into the relationship between CNN object detection systems and the human visual system, as well as the appropriateness of different attentional mechanisms for different visual tasks.

In Chapter 6, we propose a novel end-to-end auditory object recognition network. Essentially, machine learning approaches to auditory object recognition have traditionally been based on engineered features such as those derived from the spectrum or cepstrum. More recently, end-to-end classification systems in image and auditory recognition systems have been developed to learn features jointly with classification, and this approach has resulted in improved classification accuracy. In this chapter, we propose a novel end-to-end deep neural network to map the raw waveform inputs to sound class labels. Our network includes an "inception nucleus" that optimizes the size of convolutional filters on the fly, and this reduces engineering efforts dramatically. Classification results compared favorably against current state-of-the-art approaches, besting them by 10.4 percentage points on

the Urbansound8k dataset. Analyses of learned representations revealed that filters in the earlier hidden layers came to perform wavelet-like transforms in order to extract features that were informative for classification.

In Chapter 7, we talk about end-to-end approaches to analyzing infant vocalizations. Acoustic analyses of infant vocalizations are valuable for research on speech development, as well as applications in sound classification. Previous studies have focused on measures of acoustic features based on theories of speech processing, such as spectral and cepstrum-based analyses. More recently, end-to-end models of deep learning have been developed to take raw speech signals (acoustic waveforms) as inputs and convolutional neural network layers to learn representations of speech sounds based on classification tasks. We applied a recent end-to-end model of sound classification to analyze a large-scale database of labeled infant and adult vocalizations recorded in natural settings outside of the lab, with no control over recording conditions. The model learned basic classifications like infant versus adult vocalizations, infant speech-related versus non-speech vocalizations, and canonical versus non-canonical babbling. The model was trained on recordings of infants ranging from 3 to 18 months of age, and classification accuracy changed with age as speech became more distinct and babbling became more speech-like. Further work is needed to validate and explore the model and dataset, but our results show how deep learning can be used to measure and investigate speech acquisition and development, with potential applications in speech pathology and infant monitoring.

Finally, we conclude this dissertation in Chapter 8.

Chapter 2

Object Detection with Sensitivity Analysis

2.1 Introduction

Deep Convolutional Neural Networks (CNNs) have been shown to be effective at image classification, accurately performing object recognition even with thousands of object classes when trained on a sufficiently rich data set of labeled images [43]. One advantage of CNNs is their ability to learn complete functional mappings from image pixels to object categories, without any need for the extraction of hand-engineered image features [70]. To facilitate learning through stochastic gradient descent, CNNs are (at least approximately) differentiable with regard to connection weight parameters.

Image classification, however, is only one of the problems of computer vision. In the task of image classification, each image has a single label, associated with the class identity of the main object in the image, and the goal is to assign correct labels in a manner that generalizes to novel images. This can be accomplished by training a machine learning classifier, such as a CNN, on a large data set of labeled images [16]. In the object localization task, in comparison, the output for a given image is not a class label but the locations of a specified number of objects in the image, usually encoded as bounding boxes. Evaluation of an object localization system generally requires ground truth bounding boxes to compare to the system's output.In comparison, the object detection task is even more difficult than the localization task, as the number of objects are not predetermined [70].

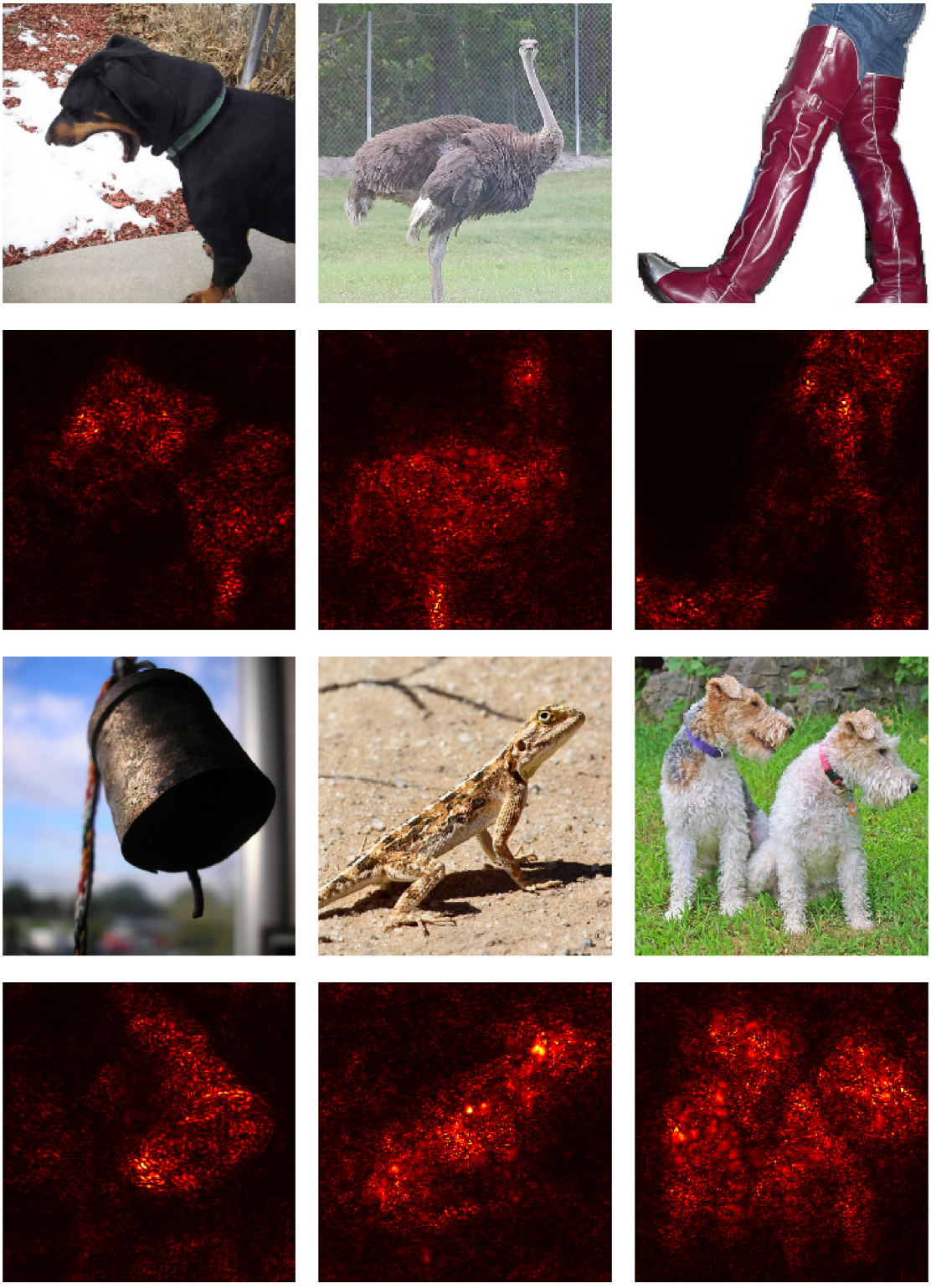

Figure 2.1: Examples of sensitivity maps, displaying the sensitivity of network internal representations to individual pixels, providing information about the locations of the main objects in the source images. Input in rows 1 and 3 are exhibiting the original images. The sensitivity analysis of the corresponding images are given in rows 2 and 4.

In this chapter, we focus on object localization, identifying the position in the image of a recognized object. As is common in the localization literature, position information is output in the form of a bounding box. Previously developed techniques for accomplishing this task generally involve searching the image for the object, considering many candidate bounding boxes with different sizes and locations, sometimes guided by an auxilliary algorithm for heuristically identifying regions of interest [70, 31, 35]. For each candidate location, the sub-image captured by the bounding box is classified for object category, with the final output bounding box either being the specific candidate region classified as the target object with the highest level of certainty or some heuristic combination of neighboring or overlapping candidate regions with high classification certainty. These approaches tend to be time consuming, often requiring deep CNN classification calculations of many candidate regions at multiple scales. Efforts to speed these methods mostly focus on reducing the number of regions considered, typically by using some adjunct heuristic region proposal algorithm [31, 61, 35].

A noteworthy alternative approach is to directly train a deep CNN to produce outputs that match ground truth localization bounding boxes, using a large image data set that provides both category and localization information for each image. It appears as if some form of this method was used with AlexNet [43], though details concerning localization, rather than image classification, are difficult to discern from the published literature. A natural approach would be to cast the learning of bounding boxes as a simple regression problem, with targets being the four coordinates that specify a bounding box (e.g., coordinates of upper-left and lower-right corners, or region center coordinates along with region width and height). It is reasonable to consider sharing early layers of a deep CNN, such as those performing convolution and max pooling, between both an image classification network and an object localization network. Indeed, taking such a multitask learning approach [9] can allow for both object category and object location training data to shape connection weights throughout the network. Thus, the deep CNN would have "two heads", one for image classification, using a classification cross-entropy loss function, and one for object localization, reducing the ℓ_2 norm between ground truth and predicted bounding box coordinates [43]. While this approach can produce a network that quickly outputs location information, extensive training on large data sets containing ground truth bounding box information is necessary to produce good generalization.

In this chapter, we introduce an approach to object localization that is both very fast and robust in the face of limited ground truth bounding box training data. This approach is rooted in the assertion that any deep CNN for image classification must contain, implicit in its connection weights, knowledge about the location of recognized objects [69]. The goal, then, is to interpret the flow of activation in an object recognition network when it is performing image classification so as to extract information about object location. Furthermore, the goal is to do this quickly. Thus, this approach aims to leverage location knowledge that is already latent in extensively trained and tuned image classification networks, without requiring an additional complex learning process for localization.

Our method makes use of the notion of a *sensitivity analysis* [75]. We propose estimating the sensitivity of the category outputs, or activation patterns at internal network layers, of an image classification CNN to variance in each input pixel, given a specific input image. The result is a numeric value for each pixel in the input image that captures the degree to which small changes in that pixel (locally, around its current value) give rise to large changes in the output category. Together, these numeric values form a *sensitivity map* of the image, encoding image regions that are important for the current classification. Our proposed measure of sensitivity is the partial derivative of activity with regard to each pixel value, evaluated for the current image. For a deep CNN that formally embodies a differentiable mapping (at least approximately) from image pixels to output categories, this partial derivative can be quickly calculated. While many tools currently exist for efficiently calculating such derivatives, we provide a simple algorithm that computes these values through a single backward pass through the image classification network, similar to that used to calculate unit error (delta) values in the *backpropagation of error* learning algorithm [64]. Thus, we can generate a sensitivity map for an image in about the same amount of time as it takes the employed image classification network to produce an output. Some example sensitivity maps are shown in Figure 2.1.

The idea of using sensitivity information, like that in our sensitivity maps, for a variety of tasks, including localization, has previously appeared in the literature [72, 92, 69]. Indeed, some of these past efforts have used more sophisticated measures of sensitivity. In this chapter, we show that even our very simple sensitivity measure can produce strong localization performance, and it can do so quickly, without any modifications to the classification network, and even for object categories on which the classification network was

not trained. The relationship of the results reported here to previously reported work is discussed further in Section 2.4.

As previously mentioned, object localization methods typically encode object location as a bounding box. Since our sensitivity maps encode location differently, in terms of pixels, we propose learning a simple linear mapping from sensitivity maps to bounding box coordinates, allowing our method to output a bounding box for each classified image. We suggest that this linear mapping can be robustly learned from a relatively small training set of images with ground truth bounding boxes, since the sensitivity maps form a much more simple input than the original images.

The primary contributions of this research may be summarized as follows:

- We propose a new general approach to performing object localization, interpreting a previously trained image classification network by performing a sensitivity analysis, identifying pixels to which the category output, or a more general internal representation, is particularly sensitive.

- We demonstrate how a linear function from the resulting sensitivity maps to object location bounding box coordinates may be learned from a relatively small set of training images containing ground truth location information.

- We provide a preliminary assessment of our approach, measuring object localization performance on the ImageNet and PASCAL VOC data sets using the VGG16 image classification CNN, showing strong accuracy while maintaining short computation times.

2.2 Object Detection via Sensitivity Analysis

2.2.1 Calculating Pixel Sensitivities in a Trained CNN

Calculating derivatives of a function of network output with regard to network parameters, such as connection weights, is a standard part of CNN training. It is common for learning in a deep CNN to involve stochastic gradient decent, which involves such derivatives. In that case, the derivatives are of an objective function with regard to connection weight values. In image classification networks, the objective function is designed to have

optima where training images are correctly classified. In the case of object localization, a similar objective function could be designed to minimize differences between output bounding box coordinates and provided ground truth bounding box coordinates, for all images in an appropriately labeled training set. For example, given N training images, stored in the matrix $\mathbf{X}$, with the ground truth 4-dimensional bounding box vector for image x_i being y_i, and $G(x_i; \mathbf{w})$ being the CNN output vector for image x_i given connection weights $\mathbf{w}$, an appropriate loss function would be:

$$\ell(\mathbf{X}, \mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \|y_i - G(x_i; \mathbf{w})\|_2^2 \tag{2.1}$$

The CNN will produce good estimates of the training image bounding boxes when this loss function is minimized with regard to $\mathbf{w}$. Network weight parameters that minimize this loss, $\mathbf{w}^*$, may be sought through stochastic gradient decent, incrementally updating $\mathbf{w}$ according to the gradient of $\ell(\mathbf{X}, \mathbf{w})$ with regard to $\mathbf{w}$. A primary drawback of this approach is that it requires a large and representative sample of images with ground truth bounding box information.

Consider that, once weights are found, the gradient of $\ell(\mathbf{X}, \mathbf{w}^*)$ with regard to $\mathbf{X}$ would provide information about the sensitivity of the bounding box loss function with regard to the pixels in the images. This gradient can be calculated as efficiently as the gradient of the loss with regard to the weights, with both depending on the gradient of $G(x_i; \mathbf{w})$ with regard to a subset of its arguments. This means that the gradient of $G(x_i; \mathbf{w}^*)$ with regard to x_i can be efficiently computed, and that gradient would capture the sensitivity of bounding box coordinates with regard to the specific pixels in image x_i. Note that this gradient can be calculated for images beyond those in the training set. Knowing which pixels in a novel image play an important role in determining the bounding box provides useful information for object localization. Using this calculation to address the object localization task makes little sense, however, as $G(x_i; \mathbf{w}^*)$ provides an estimate of object location without a need to consider pixel sensitivity.

Rather than training a deep CNN to output bounding boxes, requiring extensive labeled data, we propose calculating the same gradient for a different network – one successfully trained to perform image classification. If we now see $G(x_i; \mathbf{w}^*)$ as the output of such an image classification network, its gradient with regard to x_i would provide information about the sensitivity of the assigned category to individual pixels. Pixels with the largest

absolute values of this derivative will, around the input x_i, produce the largest changes in the classification decision of the CNN. This can be seen as one measure of how important pixels are for classifying the object in the image. Consider that the object class output is not immediately affected by changes to pixels with a derivative of zero.

The calculation of this gradient can be performed efficiently as a single "backward pass" through the classification network. This is well illustrated by considering the case of a simple layered backpropagation network [64] in which the "net input" of unit i, η_i, is a weighted sum of the activations of units in the previous layer, and the activation of unit i is $g(\eta_i)$, where $g(\cdot)$ is the unit activation function. In this case, we can define a sensitivity value for each unit, s_i, as the derivative of the network output with regard to η_i. Using the chain rule of calculus, it is easy to show that the sensitivity of an output unit is $g'(\eta i)$, and, for units in earlier layers the gradients are computed as follows:

$$s_i \;=\; g'(\eta_i) \sum_k \; w_{ki} \; s_k \tag{2.2}$$

where k iterates over all units in the immediately downstream layer from unit i and $\mathbf{w}_{ki}$ is the connection weight from unit i to unit k. This calculation may be performed, layer by layer, from outputs to inputs, until s_i values for each pixel input unit are available.

This demonstrates how efficiently pixel sensitivity values can be calculated for a given classified image. Of course, there are currently a variety of software packages that include tools for calculating gradients. In the evaluation of our approach in Section 3.4, we report results using the tools provided by TensorFlow [1].

2.2.2 Sensitivity of the Attention Map.

We have proposed using a previously trained image classification network as a source of information about object location, focusing on the gradient of the network output with regard to image pixels. It is interesting to note that it might not be necessary to perform the sensitivity calculation using the full classification network. There is a growing body of research that suggests that, in a well trained image classification CNN, the features that are extracted at the "attention map" layer (i.e., the output of the last convolutional layer) tend to be generally useful for learning a variety of image analysis tasks [57, 18]. Inspired by these results, we have investigated the possibility of substituting the gradient

of the classifier output with regard to pixels with the gradient of the attention map with regard to pixels. This avoids calculations involving final fully connected layers and any classification softmax layer. Generating image sensitivity maps from the attention map layer is slightly faster than our original proposal, but, more importantly, it is possible that general knowledge about object location might be found in the attention map, and using the attention map as the basis of the sensitivity map might actually generalize beyond the categories on which the image classiciation CNN was trained. We have not yet done a formal comparison of these two approaches to constructing the sensitivity map, but example results using both approaches are reported in Section 3.4. When computing these gradients, we refer to the aggregated values of the last convolutional layer as the Gestalt Total (GT), which is computed as follows:

$$GT = \frac{1}{H \times W \times C} \sum_{i,j,k} A_n(i,j,k) \tag{2.3}$$

where A_n is the activation map of the last convolution layer, H, W and C are the height, width, and channels of the last convolution layer.

2.2.3 Aggregating Across Color Channels

The sensitivity map calculations that have been described, so far, provide a scalar sensitivity value for each *input* to the image classification deep CNN. Color images, however, are regularly provided to such networks using multiple inputs per image pixel, often encoding each pixel over three color channels. Thus, the gradient calculation will actually produce three sensitivity values for each pixel. Since we hope to produce a sensitivity map that focuses in a general way on location information, it seems reasonable to aggregate the three sensitivity values into one. Since the direction of the sensitivity relationship with the class output is irrelevant, a good first step is to take the absolute value of each derivative. Given that dependence on even a single color channel suggests that a pixel is important for identifying the object, an argument can be made that a pixel should be labeled with the maximum of the three absolute derivatives. Alternatively, it could be argued that all color channels should be taken into account when producing the sensitivity map, in which case it might be better to average the three absolute derivatives. We have explored both of these aggregation methods, with results appearing in Section 3.4.

2.2.4 Learning to Produce Bounding Boxes

Object localization algorithms typically output the four coordinates of a bounding box to communicate the location of the target object. Such a bounding box is not intrinsic to a sensitivity map, however. Heuristic techniques could be used to identify a rectangular region that captures the majority of the high sensitivity pixels, while avoiding low sensitivity pixels, but we have taken a different approach. We have opted to learn a linear mapping from sensitivity maps to bounding box coordinates, using training images with ground truth location information.

It is important to note that learning this mapping is not the same as learning to map from the original images to bounding box coordinates, as has been done in some other object localization systems. Sensitivity maps contain much less information than the original images, so using the sensitivity maps as inputs both reduces the dimensionality of the input to this mapping and makes for a more simple functional relationship between pixels and bounding box coordinates. We expect that this simplification will allow the mapping to bounding box coordinates to be successfully learned using a far smaller set of training images labeled with ground truth object locations. Indeed, we expect that a simple linear mapping could perform well.

Formally, we define the parameters of the linear mapping to the four bounding box coordinates as a $4 \times M$ matrix, $\hat{W}$, (where M is the number of pixels in an image) and a 4-dimensional vector of "bias weights", $\hat{w}$. Given a sensitivity map, s, the output is $(\hat{W}s + \hat{w})$. Given a training set of N images, the mapping is found by minimizing the following objective function with regard to $\hat{W}$ and $\hat{w}$:

$$\frac{1}{N}\sum_{i=1}^{N}\frac{1}{4}\sum_{j=1}^{4}\|B_{i,j} - (\hat{W}s_i + \hat{w})\|_2^2 \tag{2.4}$$

where s_i is the sensitivity map for the i^{th} image, and $B_{i,j}$ is the j^{th} coordinate of the bounding box for the i^{th} image. This learning process amounts to four independent linear regression problems, which can be solved efficiently.

Once learned, mapping from sensitivity maps to bounding box coordinates can be done very quickly. With sensitivity map formation requiring only a single backward pass through

the image classification network, the whole process – from image, to classification, to sensitivity map, to bounding box – can be performed in little more than twice the time it takes for the network to do object recognition.

2.3 Results

Data Sets & Performance Measures. We evaluated our proposed method for object localization on two challenging data sets: the PASCAL VOC 2007 [26] data set and the ImageNet 2012 [16] data set. The PASCAL VOC 2007 data set was selected due to its use in the existing object localization literature. The ImageNet data set is one of the largest publicly available data sets. It also contains many images annotated with ground truth bounding boxes.

We followed the literature with regard to the evaluation criterion applied to our method, using *CorLoc*, which has been used for weakly supervised localization. The CorLoc metric is defined as the percentage of images in a data set that are correctly localized based on the PASCAL criterion, in which a given localization is considered correct if and only if the *intersection over union (IOU)* area of the predicted and ground truth bounding boxes is greater than one half:

$$IOU = \frac{area(\beta_p \cap \beta_{gt})}{area(\beta_p \cup \beta_{gt})} > 0.5 \tag{2.5}$$

. . . where β_p is the predicted bounding box and β_{gt} is the ground truth bounding box [78].

Pre-Trained Image Classification Deep CNN. To demonstrate that our approach works with an image classification deep CNN that was in no way specialized for our localization method, we opted to use a publicly available VGG16 network. This network provides ImageNet object classes as output, allowing us to calculate sensitivity maps based on the network classification when examining ImageNet data. For the PASCAL VOC 2007 data set, we used the previously described method of calculating derivatives based on the attention map of VGG16, since there is not consistent class correspondnce between the PASCAL VOC 2007 classes and the classes on which VGG16 was trained. To produce sensitivity maps for the PASCAL VOC 2007 data set, we aggregated across color channels by using the maximum absolute derivative across the three inputs for each pixel. For the ImageNet data set, we averaged the absolute derivatives across the three inputs in order to produce

pixel sensitivity values.

2.3.1 Experimental Setup

For generating sensitivity maps, we have used a pretrained VGG16 network, and we have used the whole network architecture while we were experimenting on the ImageNet dataset, otherwise we have removed the last 3 fully connected layers and computed the Gestalt Total at the last convolution layer. The derivatives in either case were computed using just one backward pass to the original pixels. For learning bounding boxes we have used the aggregated sensitivity maps as an input. To learn the mapping from sensitivity maps to bounding box coordinates, we performed linear regression using stochastic gradient decent. Updates were performed in batches of 2,048. The learning rate was initialized to 0.1 and decayed by a factor of 10 every 10,000 iterations. The experiment was run on 1 GPU for 4 days.

2.3.2 Performance on PASCAL VOC 2007

The full PASCAL VOC 2007 data set includes 12,608 training set images and an equal number of testing set images [26]. Each image contains an object of 1 of 20 different categories. We applied our object localization method to this full data set. Table 4.1 compares the localization performance of our method with that of other approaches. Note that our method, while being very fast, outperforms the comparison algorithms.

Examples of the bounding boxes selected by our method, compared to ground truth, for all 20 classes in the PASCAL VOC 2007 data set are shown in Figure 2.2. Qualitatively, it appears as if our approach is most accurate when there is a single target object with little crowding. However, if the target object is small and in a crowded region of the image, performance is less reliable.

Several of the comparison methods display better localization performance than our approach, but it is important to keep in mind that the comparison cases had some important advantages, including taking the time to use a sliding window and access to the class labels on which the network was trained. Recall that our sensitivity maps were produced, in this

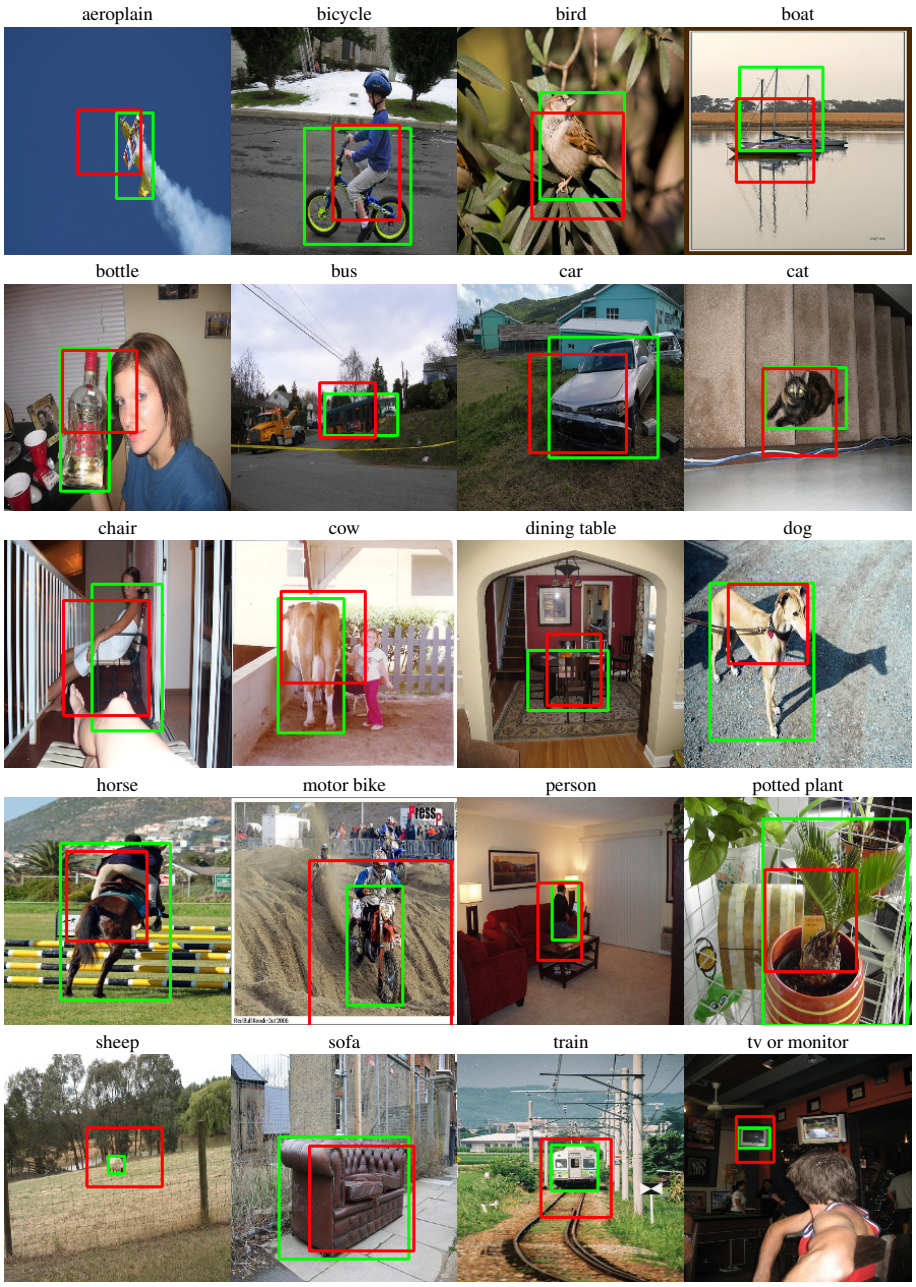

Figure 2.2: Results of the proposed method on the PASCAL VOC 2007. Each column shows three examples in one class. The green boxes are the ground truth, and the red ones are the predicted bounding boxes.

case, by calculating the sensitivity of the network attention map activity to pixel values. Thus, this comparison illustrates trade-offs between speed, performance, and generalization. To illustrate that sensitivity maps can be used without learning a linear mapping from the maps to bounding box coordiantes, we also examined a heuristic method for producing bounding boxes from maps. We used a Gaussian smoothing filter to smooth out the sensitivity maps, and then we picked up the top 20% of the pixels, heuristically drawing the bounding box so as of surround those pixels, as other researchers have done before [92, 69]. We noticed that this heuristic approach could damage the mean CorLoc by 3% in our best observations. However, this process highly depends on the smoothing parameter σ . The obtained results from different σ values are reported in Table 2.2.

2.3.3 Performance on ImageNet

ImageNet is a large image data set that has been systematically organized by object category [16]. We executed a large scale evaluation of our approach by using all images in ImageNet that are annotated with ground truth localization information. This subset contains 300,916 images involving 478 object classes. We divided this data set into a training set, a test set, and a validation set by sampling without replacement (i.e., the intersection between each pair of the three sets was empty). There were 225,687 images (75%) in the training set, and there were 45,137 images in each of the other two sets.

We compared the performance of our approach with two methods discussed in Tang et al. [78] for which ImageNet results are explicitly reported: Top Objectiveness Box & Co-Localization. Also, we noted that many images in this data set presented the target object in the middle of the image, providing a bias that could be leveraged by learned localization systems. Thus, as a baseline of performance, we calculated the CorLoc performance for

Table 2.1: PASCAL VOC 2007 Test Detection Results. The proposed method performed favorably against the state of the art methods.

Method	MeanCL	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
[74]	30.2	45.8	21.8	30.9	20.4	5.3	37.6	40.8	51.6	7.0	29.8	27.5	41.3	41.8	47.3	24.1	12.2	28.1	32.8	48.7	9.4
[71]	36.2	67.3	54.4	34.3	17.8	1.3	46.6	60.7	68.9	2.5	32.4	16.2	58.9	51.5	64.6	18.2	3.1	20.9	34.7	63.4	5.9
[34]	38.8	56.6	58.3	28.4	20.7	6.8	54.9	69.1	20.8	9.2	50.5	10.2	29.0	58.0	64.9	36.7	18.7	56.5	13.2	54.9	59.4
OM [44]	31.8	50.4	30	34.6	18.2	6.2	39.3	42.2	57.3	10.8	29.8	20.5	41.8	43.2	51.8	24.7	20.8	29.2	26.6	45.6	12.5
PBRM [11]	36.6	50.3	42.8	30.0	18.5	4.0	62.3	64.5	42.5	8.6	49.0	12.2	44.0	64.1	57.2	15.3	9.4	30.9	34.0	61.6	31.5
Sensitivity Maps	**40.1**	63.8	55.1	41.2	23.3	34.2	58.6	72.7	36.9	23.3	49.7	11.5	29.6	50.1	65.9	11.8	42.2	39.7	18.1	51.0	41.2

Table 2.2: Average CorLoc Performance on Pascal VOC 2007 using the heuristic bounding box (σ is the smoothing parameter.)

σ	CorLoc
10	27.2%
20	38.4%
30	32.5%

a system that blindly offered the same bounding box in the middle of the image, with average size, for every input. The results are shown in Table 2.3. Once again, note the relatively high accuracy performance of our efficient method. Also note that the baseline was comfortingly low. As might be expected, performance varies with class. Our algorithm appears to do well on some objects, such as balls and dogs. One might suspect that failures arise in the linear mapping from sensitivity maps to bounding box coordinates, but a perusal of the sensitivity maps, themselves, suggests that the pixel sensitivity values vary in utility across different object categories. Still, our method performs fairly well across the classes. Note that the IOU does not fall below 0.62 for any class. This suggests that, while some individual images may be problematic, the overall performance for each class is quite good. This universally strong class-specific performance is also displayed in Table 2.3.

2.3.4 Methods Analysis

The sensitivity analysis approach gives us the sensitivity of single pixels in all channels in the RGB images. Since we are in need of locations, we need to aggregate among channels. We proposed two methods, an average function and a maximum function. The first approach calculates the sensitivity as the average among channels, and the second method uses the maximum absolute derivatives among channels. We didn't notice a significant dif-

Table 2.3: CorLoc Performance on ImageNet (478 Classes)

Method	Average CorLoc
Constant Center Box Baseline	12.34%
Top Objectiveness Box	37.42%
Co-Localization	53.20%
Sensitivity Maps	**68.76%**

ference between these two methods in localization performance. The only insight is that generating sensitivity maps based on the average function is a bit smoother in a visual sense than the maximum function. The CorLoc between average and maximum aggregation functions on the ImageNet dataset are 68.7 and 67.9 respectively, and the results for these two aggregation operators on the PASCAL VOC dataset is 39.2 and 40.1, respectively.

2.4 Summary

We have presented an approach to object localization based on performing a sensitivity analysis of a previously trained image classification deep CNN. Our method is fast enough to be used in online applications, and it demonstrates accuracy that is superior to some methods that are much slower. It is likely that even better accuracy could be had by incorporating sensitivity analysis information into a more sophisticated bounding box estimator.

As previously noted, the idea of using sensitivity information has appeared in previously published work. There are ways in which the results reported in this work are distinct, however. We have moved beyond visualization of network function using sensitivity (or *saliency*) [72] to performing direct comparisons between different methods on the localization task. We have shown that using a fast and simple measure of sensitivity can produce comparable performance to that of much slower methods. Our approach produces good generalization without modifying the classification network, as is done in *Class Activation Mapping (CAM)* [92]. With our PASCAL VOC 2007 results, we have shown that our approach can successfully be applied to attention maps, even when the image contains objects belonging to a class on which the classification network was not trained, distinguishing it from *Grad-CAM* [69]. In short, we have demonstrated the power of a simple sensitivity measure for performing localization.

Note that our approach may be used with image classifiers other than CNNs. The proposed sensitivity analysis can be conducted on any differentiable classifier, though performance will likely depend on classifer specifics. Indeed, at a substantial time cost, even a black box classifier could be approximately analyzed by making small changes to pixels and observing the effects on activation patterns.

The proposed approach is quite general. For example, we could apply sensitivity analysis to deep networks trained on other tasks, with the goal of interpreting network perfor-

mance on the current input in a useful way. Thus, we see a potentially large range of uses for sensitivity analysis in neural network applications.

Chapter 3

Object Detection with Selective Attention

3.1 Introduction

Focus matters. In order to accurately detect objects in an image, we need to know which parts of the image are important and then focus on those parts to find objects of interest. Many approaches to *selective attention* in artificial neural networks are computationally expensive, however, limiting their utility for online applications. The computer vision task of object detection involves locating and classifying all of the relevant objects in an image. This is a challenging problem that has been explored by number of researchers [61, 31, 32, 58]. Since there is rarely *a priori* information about where objects are located in an image, most approaches to object detection conduct search over image regions, seeking objects of interest with different sizes and at different scales. For example, "region proposal" frameworks, like Faster-RCNN [61], need to pass a large number of candidate image regions through a deep network in order to determine which parts of the image contain the most information concerning objects of interest. An alternative approach involves one shot detectors, like Single Shot Detectors (SSD) [47] and You Only Look Once (YOLO) [58]. These use networks to examine all parts of the image via a tiling mechanism. For example, YOLO conducts a search over potential combinations of tiles. In a sense, most off-the-shelf object detection algorithms distribute attention to all parts of the image equally, since there is no prior information concerning where objects of interest

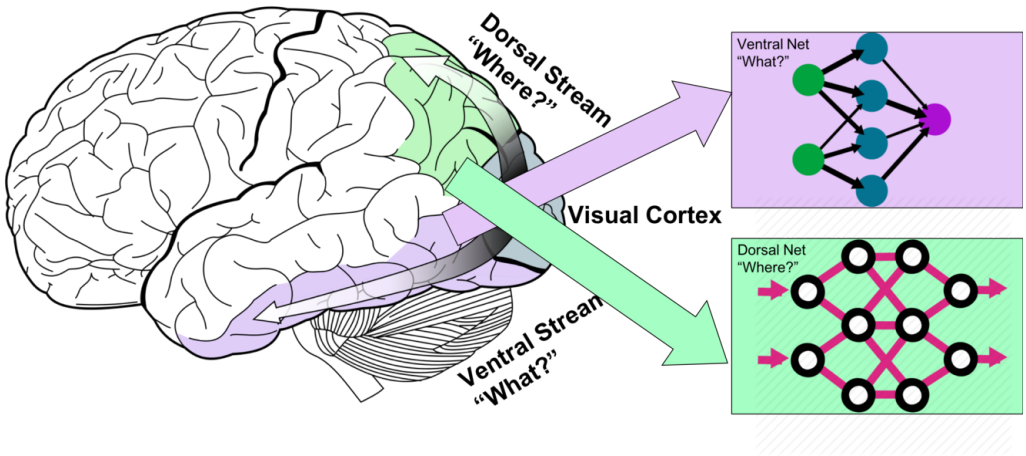

Figure 3.1: Primary visual cortex and two processing streams. The "ventral stream" projects into the temporal lobe, and the "dorsal stream" extends into the parietal lobe. These interacting pathways inspire the use of a "Ventral Net" and a "Dorsal Net".

might be found.

In contrast, there is much evidence that the human visual system is equipped with a useful spatial **selective attention** capability that guides processing to relevant parts of the scene while effectively ignoring irrelevant parts [17]. Some theories of spatial attention focus on divergent but interacting neural pathways in the brain's visual system. Projecting from primary visual cortex in the occipital lobe, two neural streams emerge, as shown in Figure 3.1. One stream, extending ventrally into the temporal lobe, appears to largely encode *what* is in the scene, recognizing objects. The other stream, passing dorsally into the parietal lobe, approximately captures information about *where* objects are located. Some computational neuroscience accounts characterize spatial attention as naturally arising from interactions between these two neural pathways [50]. Inspired by this theory of attention, we propose a novel object detection framework that integrates two networks that reflect the two pathways of the human visual system. The *Ventral Net* uses object classification information to guide attention to relevant image regions, and the *Dorsal Net* uses this attentional information to accurately localize and identify objects in the scene. Together, these two components form a *Ventral-Dorsal Neural Network* (VDNet).

Our object detection framework can be seen as arising from the marriage of **attention based object detection** [86, 85] and **supervised object detection** [58, 61]. The general

approach is to use ideas from attention based object detection to quickly identify parts of the image that are irrelevant, with regard to objects of interest, using this information to guide selective attention. Focusing further supervised processing on important image regions is expected to both allow for the allocation of computational resources in a more informed manner and produce more accurate object detection by removing potentially distracting background pixels. Our selective attention process is based on *Sailency Detection* in activation maps, such as the methods described in Chapter 2. Guided by the resulting attentional information, we use supervised object detection to achieve high object detection performance. Thus, this framework can be seen as a combination of a top-down attention based model and the bottom-up detection based approach. As far as we know, the proposed approach is novel. Moreover, our method performs favorably against state-of-the-art object detection approaches by a large margin. In overview, the work described in this chapter makes these contributions:

- Our approach uses a top-down saliency analysis to identify irrelevant image regions, introducing a selective attention mechanism that masks out noise and unimportant background information in the image.
- Our measure of top-down saliency involves a sensitivity analysis of a trained object classification network with regard to the Gestalt Total (GT) activation produced by the network.
- Our method performs very favorably in comparison to state-of-the-art approaches, demonstrating an improvement of up to 8% (mAP) in comparsion to the best of those approaches.

3.2 Related Work

State-of-the-art object detection approaches are based on deep CNNs [61, 58, 59, 35]. Leading object detection algorithms can be roughly divided into two categories: attention based object detection & supervised object detection.

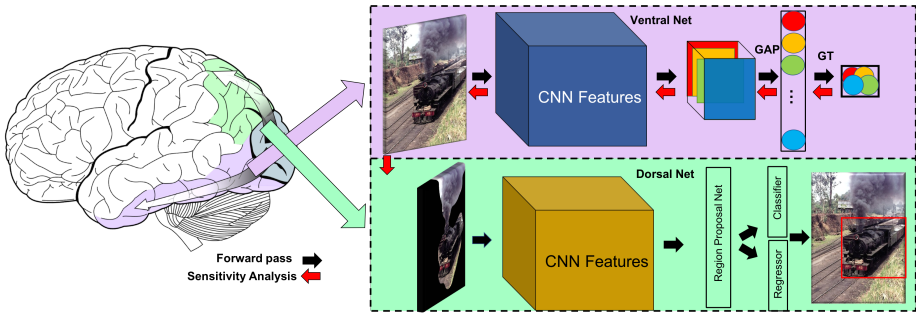

Figure 3.2: The Ventral-Dorsal Network (VDNet) for Object Detection. The Ventral Net filters out irrelevant parts of the image based on a sensitivity analysis of the Gestalt Total (GT) activation. The Dorsal Net then searches for objects of interest in the remaining regions.

3.2.1 Attention Based Object Detection

Attention based object detection methods depend on a set of training images with associated class labels but *without* any annotations, such as bounding boxes, indicating the locations of objects. The lack of ground truth bounding boxes is a substantial benefit of this approach, since manually obtaining such information is costly.

One object detection approach of this kind is the Class Activation Map (CAM) method [92]. This approach is grounded in the observation that the fully connected layers that appear near the output of typical CNNs largely discard spatial information. To compensate for this, the last convolutional layer is scaled up to the size of the original image, and Global Average Pooling (GAP) is applied to the result. A linear transformation from the resulting reduced representation to class labels is learned. The learned weights to a given class output are taken as indicating the relative importance of different filters for identifying objects of that class. For a given image, the individual filter activation patterns in the upscaled convolutional layer are entered into a weighted sum, using the linear transformation weights for a class of interest. The result of this sum is a Class Activation Map that reveals image regions associated with the target class.

The object detection success of the CAM method has been demonstrated, but it has also inspired alternative approaches. The work of Selvaraju et al. [69] suggested that Class Activation Maps could be extracted from standard image classification networks without any modifications to the network architecture and additional training to learn filter weights.

The proposed Grad-CAM method computes the gradients of output labels with respect to the last convolutional layer, and these gradients are aggregated to produce the filter weights needed for CAM generation. This is an excellent example of saliency based approaches that interpret trained deep CNNs, with others also reported in the literature [93, 90, 91].

As previously noted, attention based object detection methods benefit from their lack of dependence on bounding box annotations on training images. They also tend to be faster than supervised object detection approaches, producing results by interpreting the internal weights and activation maps of an image classification CNN. However, these methods have been found to be less accurate than supervised object detection techniques.

3.2.2 Supervised Object Detection

Supervised object detection approaches require training data that include both class labels and tight bounding box annotations for each object of interest. Explicitly training on ground truth bounding boxes tends to make these approaches more accurate than weakly supervised methods. These approaches tend to be computationally expensive, however, due to the need to search through the space of image regions, processing each region with a deep CNN. Tractability is sought by reducing the number of image regions considered, selecting from the space of all possibe regions in an informed manner. Methods vary in how the search over regions is constrained.

Some algorithms use a **region proposal based framework**. A deep CNN is trained to produce both a classification output and location bounding box coordinates, given an input image. Object detection is performed by considering a variety of rectangular regions in the image, making use of the CNN output when only the content in a given region is presented as input to the network. Importantly, rather than consider all possible regions, the technique depends on a *region proposal algorithm* to identify the image regions to be processed by the CNN. The region proposal method could be either an external algorithm like Selective Search [81] or it could be an internal component of the network, as done in Faster-RCNN [61]. The most efficient object detection methods of this kind are R-CNN [32], Fast-RCNN [31], Faster-RCNN [61], and Mask-RCNN [35]. Approaches in this framework tend to be quite accurate, but they face a number of challenges beyond issues of speed. For example, in an effort to propose regions containing objects of one

of the known classes, it is common to base region proposals on information appearing late in the network, such as the last convolutional layer. The lack of high resolution spatial information late in the network makes it difficult to detect small objects using this approach. There are a number of research projects that aim to address this issue by combining low level features and high level ones in various ways [93, 45].

Rather than incorporating a region proposal mechanism, some supervised methods perform **object detection in one feed-forward pass**. A prominent method of this kind is YOLO, as well as its extensions [58, 59]. In this approach, the image is divided into tiles, and each tile is annotated with anchor boxes of various sizes, proposing relevant regions. The resulting information, along with the image tiles, are processed by a deep network in a single pass in order to find all objects of interest. While this technique is less accurate than region proposal approaches like Faster-RCNN, it is much faster, increasing its utility for online applications.

A comparison of these two general approaches to object detection displays a clear trade-off between accuracy and computational cost (speed). This gives rise to the question of whether this trade-off can be avoided, in some way. In this chapter, we propose combining attention based object detection and supervised object detection in a manner that leverages the best features of both. Inspired by the human visual system and its interacting ventral and dorsal streams, we propose using a fast attention based approach to mask out irrelevant parts of the original image. The resulting selective attention greatly reduces the space of image regions to be considered by a subsequent supervised network.

3.3 Object Detection with Selective Attention

3.3.1 Overview

Human vision regularly employs selective attention. Without introspective awareness of the process, our brains appear to quickly focus visual processing on important parts of a scene, apparently masking out unimportant parts. Our proposed object detection framework is inspired by this phenomenon. It uses a kind of selective attention to quickly eliminate distractions, backgrounds, and irrelevant areas from the image before object detection is performed, providing a sharper focus to the process. VDNet incorporates two distinct

networks: one for quickly guiding selective attention and a second for classifying and localizing objects of interest.

The *Ventral Net* is used to determine how to apply selective attention to the image. Ventral Net is a deep CNN that first extracts convolutional features from the input image and then aggregates the results into a *Gestalt Total* (GT) output. A sensitivity analysis is then performed, identifying those pixels in the original image that have the greatest immediate influence on the GT value. A simple smoothing operation translates the clouds of salient pixels into relevant regions in the image. The result is a guide for selective attention in the original image space. It provides an indication of which parts of the image are important and which are irrelevant.

The *Dorsal Net* performs supervised object detection, but it does so on a modified version of the image in which irrelevant regions, as determined by the *Ventral Net*, have been simply masked out. Selective attention speeds the region proposal process by restricting consideration to unmasked portions of the image. The result is class labels and bounding boxes for the objects of interest in the image.

In the remainder of this section, our new framework is described in detail. VDNet is illustrated in Figure 3.2. The following notation is used: $X \in \mathcal{R}^{m \times n \times c}$ is the image input of the system, where m, n, and c are the width, height, and number of channels, respectively; an associated ground truth class label is z, and $b \in \mathcal{R}^4$ is the associated ground truth bounding box coordinates of the object in the input image.

3.3.2 Ventral Net

Ventral Net is responsible for guiding selective attention. It is built upon the convolutional layers of a deep CNN that has been trained to perform image classification. The last convolutional layer is used to calculate the Gestalt Total (GT) activation for the current input.

For a given image, let $f_k(x, y)$ denote the activation of filter k in the last convolutional layer at spatial location (x, y). For filter k, the Global Average Pooling (GAP) value is defined as:

$$F^k = \sum_{x,y} f_k(x, y) \tag{3.1}$$

The GT simply aggregates these values across all of the filters:

$$GT = \sum_k F^k \tag{3.2}$$

While this simple scalar value might seem to carry little useful information, it provides us with a simple way to identify the source pixels that have the greatest influence on the final convolutional layer activity, in a local sense. These pixels are identified through a *sensitivity analysis*, measuring the degree to which changes in an input pixel affect the GT. This is easily computed as follows:

$$S = \frac{\partial\, GT}{\partial\, X}\,\Bigg|_{X=I_i} \tag{3.3}$$

where X is the network input and $I_i \in \mathcal{R}^{m\times n\times c}$ is the i^{th} image in the dataset. The result of this sensitivity analysis is $S \in \mathcal{R}^{m\times n\times c}$. Since we are interested in all the important pixels in the image, we work with the absolute values of S. Just as methods for stochastic gradient descent can involve the calculation of gradients of the output error, for a given input, with regard to network weights using a single forward pass and a single backward pass through the network, the value of S can be quickly calculated for each image in the dataset.This process was described in some detail in Chapter 2.

Derivatives are calculated for all of the inputs to the network, which typically include three channels per pixel (i.e., RGB). Since the purpose of Ventral Net is to guide spatial attention, a single measure of relevance for each pixel location, not each channel, is needed. Thus, some way to aggregate across channels is needed. We have considered two different methods of aggregation. The first involves averaging derivative values across channels:

$$\hat{S}_{x,y} = \frac{1}{k}\sum_k S_{x,y,k} \tag{3.4}$$

where k is the number of channels in the input image (i.e., 3 for RGB) and $\hat{S}$ is the $\mathcal{R}^{m\times n}$ result of aggregating derivatives. An alternative aggregation method is to use the maximum derivative across channels:

$$\hat{S}_{x,y} = \max_k(S_{x,y,k}) \tag{3.5}$$

The resulting $\hat{S}$ provides a measure of relevance at the individual pixel level. (The evaluation experiments reported in this chapter all used the averaging approach to aggregation.)

In order to translate this information into larger regions of relevance, we smooth $\hat{S}$ by convolving it with a Gaussian filter, resulting in $\tilde{S}$. To extract distinct regions from the resulting smoothed attention map, pixels need to be classified as relevant or irrelevant. For simplicity, this is done by setting to zero any value in $\tilde{S}$ below the mean over all pixels and setting the other values in $\tilde{S}$ to one. The resulting binary mask is duplicated across the number of channels in the original image, aligning the dimensionality of the mask and the image.

The mask is applied to the original image through simple element-wise multiplication. Formally:

$$\hat{I} = I \odot \tilde{S} \tag{3.6}$$

where I is the original image and $\hat{I}$ is the masked image. This modified image then becomes the input to Dorsal Net.

3.3.3 Dorsal Net

Reducing the original image, I, to one in which irrelevant regions are masked out, $\hat{I}$, substantially reduces the space of candidate regions to consider during object detection. In Dorsal Net, the masked image is provided as input to a deep CNN trained to propose regions of interest with anchor boxes, process the contents of those regions, and output both class labels and bounding box coordinates. This is done using the methods of Faster-RCNN [61]. Dorsal Net is trained using a dataset of images that are annotated with both ground truth class labels and ground truth bounding boxes. Network parameters are selected so as to minimize a combination of the classification loss and the regression loss arising from the output of bounding box coordinates. The complete objective function is:

$$\begin{aligned} L(p_i, t_i) = & \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) \\ & + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*). \end{aligned} \tag{3.7}$$

where i is the index of an anchor box appearing in the current training mini-batch and p_i is the predicted probability of anchor i containing an object of interest. The ground truth label p_i^* is 1 if anchor i is positive for object presence and it is 0 otherwise. The predicted bounding box is captured by the 4 element vector t_i, and t_i^* contains the coordinates of the

ground truth bounding box associated with a positive anchor. The two components of the loss function are normalized by N_{cls} and N_{reg}, and they are weighted by a balancing parameter, λ. In our current implementation, the classification loss term is normalized by the mini-batch size (i.e., $N_{cls} = 32$) and the bounding box regression loss term is normalized by the number of anchor locations (i.e., $N_{reg} \approx 2,400$). We set $\lambda = 10$, making the two loss terms roughly equally weighted.

It is worth noting that our general approach could easily support the implementation of Dorsal Net using a wide variety of alternative algorithms. The only requirement is that the object detection algorithm must be able to accept masked input images. For the results presented in this chapter, we have used a leading region proposal based approach due to the high accuracy values reported for these methods in the literature. Having Ventral Net reduce the number of proposed regions is expected to speed the object detection process, and it may also improve accuracy by removing from consideration irrelevant portions of the image.

The general architecture of VDNet appears in Figure 3.2. Pseudocode for the proposed VDNet method is provided in Algorithm 1. An experimental evaluation of VDNet appears next.

3.4 Experimental Evaluation

3.4.1 Experiment Design and Implementation

We evaluated VDNet on PASCAL VOC 2007 [26], PASCAL VOC 2012 [27], and on a dataset of Yearbook images.

The PASCAL VOC 2007 dataset has 20 classes and $9,963$ images which have been equally split into a training/validation set and a test set. The PASCAL VOC 2012 dataset contains $54,900$ images from 20 different categories, and it has been split approximately equally into a training/validation set and a test set. An additional examined dataset is a private Yearbook dataset owned by Ancestry.com Operations Inc., used with permission. This dataset has more than 75 million yearbook page images. We trained a separate model to detect portraits and group photos in these images.

For PASCAL VOC 2007, we conducted training on the union of the VOC 2007 trainval

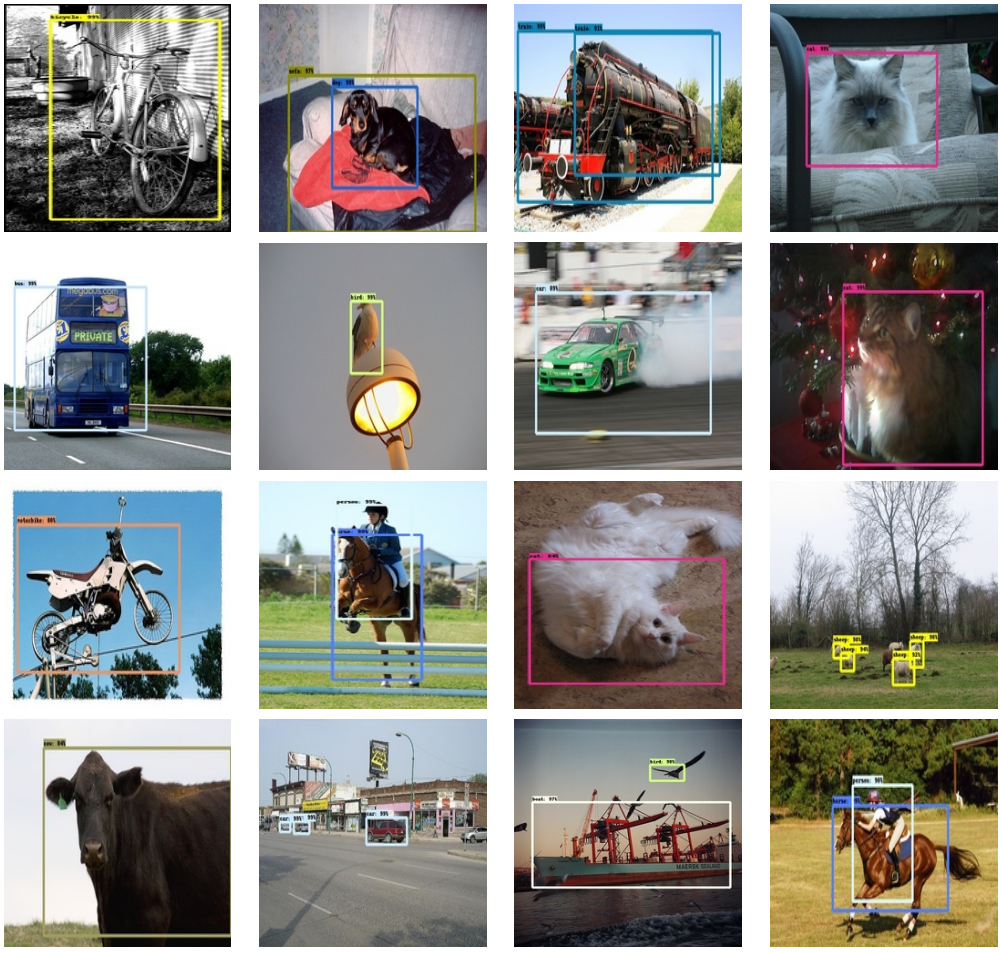

Figure 3.3: VDNet Performance on Some PASCAL VOC 2007 Validation Images

Algorithm 1: Ventral-Dorsal Net Pseudocode

Input: $\mathbf{I}, z, b$, a labeled RGB image;
Output: $\mathbf{I}$, an RGB image with labeled bounding boxes around objects of interest;
VentralNet;
    Forward pass $\mathbf{I}$ though the convolutional layers to get F.;
    Compute GT based on Equations 3.1 and 3.2.;
    Calculate the sensitivity of GT with respect to the input via Equation 3.3.;
    Aggregate the sensitivity maps for spatial locations using Equation 3.4 or Equation 3.5.;
    Smooth the result by convolving the aggregated map with a Gaussian filter. ;
    Threshold the smoothed map to produce a binary mask.;
    Mask the input image using Equation 3.6, producing $\hat{I}$. ;
DorsalNet;
    Forward pass $\hat{I}$ though a pretrained network to obtain the convolution features.;
    Run the region proposal network using the resulting feature map.;
    Pool regions of interest (ROIs).;
    Pass pooled ROI to the fully connected layers;
    Minimizing the loss function given in Eq. 4.10 ;

set and the VOC 2012 trainval set, and we evaluated the result using the VOC 2007 test set. (This regimen is standard practice for these datasets.) For PASCAL VOC 2012, we performed training on its trainval set, and we evaluated the result on its test set. For the Yearbook dataset, we trained on 1 million images, and we evaluated the results on 10,000 test set images.

To evaluate performance, we used the standard mean average precision (mAP) measure. For all datasets we report mAP scores using IoU thresholds at 0.5.

For networks with 224×224 image inputs, using PASCAL VOC, we trained the model with a mini-batch size of 16 due to GPU memory constraints. We started the learning rate at 3×10^{-4} for the first $900,000$ epochs. We decreased it to 3×10^{-5} until epoch $1,200,000$. Then, we decreased it to 3×10^{-6} until epoch $2,000,000$. In all cases, we used a momentum optimizer value of 0.9.

Figure 3.4: Some examples of selective attention image masking based on Ventral Net sensitivity analyses. The first column shows original images, and the second column displays the results of Ventral Net based masking.

Method	Network	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster [61]	VGG	73.2	76.5	79	70.9	65.5	52.1	83.1	84.7	86.4	52	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83	72.6
ION [5]	VGG	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87	54.4	80.6	73.8	85.3	82.2	82.2	74.4	47.1	75.8	72.7	84.2	80.4
Faster [36]	Residual-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72
MR-CNN [29]	VGG	78.2	80.3	84.1	78.5	70.8	68.5	88	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85	76.4	48.5	76.3	75.5	85	81
R-FCN [14]	Residual-101	80.5	79.9	87.2	81.5	72	69.8	86.8	88.5	89.8	67	88.1	74.5	89.8	90.6	79.9	81.2	53.7	81.8	81.5	85.9	79.9
SSD300 [47]	VGG	77.5	79.5	83.9	76	69.6	50.5	87	85.7	88.1	60.3	81.5	77	86.1	87.5	83.9	79.4	52.3	77.9	79.5	87.6	76.8
SSD512 [47]	VGG	79.5	84.8	85.1	81.5	73	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79	86.6	80
DSSD321 [28]	Residual-101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5	78.7	86.7	88.7	86.7	79.7	51.7	78	80.9	87.2	79.4
DSSD513 [28]	Residual-101	81.5	86.6	86.2	82.6	74.9	62.5	89	88.7	88.8	65.2	87	78.7	88.2	89	87.5	83.7	51.1	86.3	81.6	85.7	83.7
STDN300 [93]	DenseNet-169	78.1	81.1	86.9	76.4	69.2	52.4	87.7	84.2	88.3	60.2	81.3	77.6	86.6	88.9	87.8	76.8	51.8	78.4	81.3	87.5	77.8
STDN321 [93]	DenseNet-169	79.3	81.2	88.3	78.1	72.2	54.3	87.6	86.5	88.8	63.5	83.2	79.4	86.1	89.3	88.0	77.3	52.5	80.3	80.8	86.3	82.1
STDN513 [93]	DenseNet-169	80.9	86.1	89.3	79.5	74.3	61.9	88.5	88.3	89.4	67.4	86.5	79.5	86.4	89.2	88.5	79.3	53.0	77.9	81.4	86.6	85.5
VDNet	Resnet-101	**86.2**	95.8	98.1	98.4	65.1	94.6	90.1	96.2	71.7	72.3	54.6	97.9	95.6	89.2	90.1	93.2	69.1	89.2	82.1	93.4.6	74.0

Table 3.1: PASCAL VOC 2007 Test Detection Results. Note that the minimum dimension of the input image for Faster and R-FCN is 600, and the speed is less than 10 frames per second. SSD300 indicates the input image dimension of SSD is 300×300. Large input sizes can lead to better results, but this increases running times. All models were trained on the union of the trainval set from VOC 2007 and VOC 2012 and tested on the VOC 2007 test set.

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
HyperNet-VGG [42]	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.97	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
HyperNet-SP [42]	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO [58]	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2
MR-CNN-S-CNN [29]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [61]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
NoC [62]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
VDNet	**73.2**	85.1	82.4	73.6	57.7	61.2	79.2	77.1	85.5	54.9	79.8	61.4	87.1	83.6	81.7	77.9	45.6	74.1	64.9	80.3	73.1

Table 3.2: PASCAL VOC 2012 Test Detection Results. Note that the performance of VDNet is about 3% better than baseline Faster-RCNN.

3.4.2 PASCAL VOC 2007

Comparative Performance Results

The results of the PASCAL VOC 2007 dataset evaluation appear in Table 3.1. For the Ventral Net, we utilized VGG19 pretrained on the ImageNet dataset. We removed the fully connected layers and softmax calculation from VGG19, and we calculated GT based on the last convolutional layer. The Ventral Net sensitivity analysis was performed on the resulting network. No fine tuning of parameters was done. For the Dorsal Net we used Resnet 101 pretrained on the ImageNet dataset as the backbone network for object detection. The model was trained on 8 GPUs hosted by Amazon Web Services (AWS) for about one week.

We compared our performance results with those reported for a variety of state-of-

the-art approaches to object detection. Our primary baseline was Faster-RCNN using a Resnet 101 network trained on PASCAL VOC 2007. As shown in Table 4.1, the selective attention process of our approach (VDNet) resulted in substantially better performance in comparison to Faster-RCNN and other methods. VDNet appears to be more accurate at detecting larger objects than smaller ones, perhaps because the region proposal network based its output on the last (lowest resolution) convolutional layer. For most of the object classes, VDNet performed better than other methods by a large margin. For some classes, however, the selective attention mechanism sometimes failed to identify the relevant parts of the image, resulting in poorer performance. Some illustrative test data examples are provided in Figures 3.3 and 3.4.

Discussion

In order to further understand the contributions of the various components of VDNet, we observed the results of focal modifications to the system.

We noticed that poor performance could frequently be traced to poor selective attention masks. By varying parameters, we found that the variance of the Gaussian filter used to smooth the sensitivity analysis map played an important role. A poor choice for the variance could result in highly inappropriate attentional masks. We found that good performance could be had on the PASCAL datasets by using a variance value between 25 and 35. It is likely, however, that this value would need to be tuned to the size and kinds of objects to be detected.

We considered performing the Ventral Net sensitivity analysis using the output class labels of the pretrained image classification network, rather than basing that analysis on the Gestalt Total (GT) activation of the last convolutional layer. It seems natural to ask for the set of pixels that contribute most to the recognition of an object of a particular class. There are a couple of reasons why we found sensitivity of GT to pixels to be a better measure. First, focusing on the last convolutional layer allowed us to produce reasonable sensitivity maps even for object classes novel to the pretrained classification network. Second, since object detection often involves scenes containing multiple objects, producing sensitivity maps based on class outputs would require the aggregation of sensitivity information for each class that might appear in the image. In the most general case, this means calculating

a separate sensitivity map for each class, increasing the execution time of the Ventral Net process by a factor equal to the number of classes.

We performed a cursory examination of the role of network depth in the Dorsal Net on object detection performance on the PASCAL VOC 2007 dataset. In general, it seems as if increasing network depth increases object detection accuracy. This observation was based on comparing three different implementations of the Dorsal Net: Inception, Resnet 50, and Resnet 101. Accuracy was much better for the deeper networks, as shown in Table 3.3.

VDNet Component	Deep Network	mAP
Dorsal Net	Inception	63.1
Dorsal Net	ResNet50	71.6
Dorsal Net	ResNet101	86.2

Table 3.3: PASCAL VOC 2007 Test Results for Different Network Architectures

3.4.3 PASCAL VOC 2012

We also measured VDNet performance on the PASCAL VOC 2012 dataset. The Ventral Net consisted of VGG19, with features for calculating GT extracted from the last convolutional layer. The Dorsal Net was initialized with parameters previously learned for the PASCAL VOC 2007 evaluation, but further training was done. For the additional training, the learning rate was initialized to 3×10^{-4} for $900,000$ epochs, and then it was reduced to 3×10^{-5} until reaching epoch $1,200,000$. The learning rate was further reduced to 3×10^{-6} until reaching $3,000,000$ epochs. The whole training process took about 14 days on 8 GPUs hosted by AWS. This resulted in a VDNet that produced comparable or better performance that state-of-the-art methods. Performance results for PASCAL VOC 2012 are shown in Table 3.2.

3.4.4 Yearbook Dataset

We performed additional evaluation experiments using a dataset of grayscale Yearbook page images, testing VDNet in a somewhat different application domain. This dataset is not publicly available, but was provided by Ancestry.com Operations Inc, which possesses

more than 75 million yearbook page images. Pictures on these pages appear at a variety of different angles, with different shapes, scales, and sizes. The task was to automatically detect and crop the portraits and the group photos from the pages.

Because of the differences in image properties, it did not make sense to use a network pretrained on the ImageNet dataset as the Ventral Net. Instead, we produced a classification network appropriate for this task. Our architecture included two convolutional layers, with 32 and 64 filters, respectively. The last convolutional layer was followed by max pooling and GAP, feeding a fully connected layer with Relu units followed by a softmax operation. Classification was done to recognize group photos and portraits in 224×224 image patches. A batch size of 32 was used, and training was conducted for $1,000$ epochs. No regularization (e.g., weight decay, drop out) was used. The learning rate was initialized to 10^{-1}, and it decayed exponentially every $1,000$ epochs. The Dorsal Net was Resnet 101.

We compared our VDNet to Faster-RCNN and discovered that VDNet exhibited a 7% improvement in accuracy.

3.5 Summary

In this chapter, we highlighted the utility of incorporating fast selective attention mechanisms into object detection algorithms. We suggested that such mechanisms could potentially speed processing by guiding the search over image regions, focusing this search in an informed manner. In addition, we demonstrated that the resulting removal of distracting irrelevant material can improve object detection accuracy substantially.

Our approach was inspired by the visual system of the human brain. Theories of spatial attention that see it as arising from dual interacting "what" and "where" visual streams led us to propose a dual network architecture for object detection. Our Ventral Net consists of a pretrained image classification network, and a sensitivity analysis of this network, focusing on the Gestalt Total (GT) activation of the final convolutional layer, is performed to provide a fast identification of relevant image regions. The Ventral Net guides selective attention in the Dorsal Net, restricting the spatial regions considered by the Dorsal Net as it performs high accuracy object detection. Our approach, VDNet, integrates attention based object detection methods with supervised approaches.

The benefits of selective attention, as implemented in VDNet, are evident in perfor-

mance results on the PASCAL VOC 2007 and PASCAL VOC 2012 datasets. Evaluation experiments revealed that VDNet displays greater object detection accuracy than state-of-the-art approaches, often by a large margin.

There are many opportunities for improving upon VDNet. One noteworthy weakness was observed in the detection of smaller objects, arising from the use of later, lower resolution, convolutional layers for the generation of region proposals. Ongoing research, including some reported in subsequent chapters, focuses on equipping VDNet with network architectures that support learning at multiple spatial scales. The idea is to perform senstivity analyses in Ventral Net based on convolutional layers throughout the classification network, combining these in a manner that integrates information from smaller spatial scales without being distracted by more fundamental image features. It is important that the result of the sensitivity analysis continues to focus on the visual properties that are closely related to the presence of objects of interest.

Finally, it is worth noting that VDNet has a very general and flexible structure. The Ventral Net and the Dorsal Net can easily be updated to incorporate the latest advancements in image classification and object detection algorithms. In this way, we see VDNet as opening many avenues for future exploration.

Chapter 4

Dense Saliency Maps

4.1 Introduction

In the previous chapters, we have proposed an object detection framework inspired by a computational neuroscience theory of human spatial attention, called Ventral-Dorsal Nets or, simply, VDNets. This framework aims to guide object detection by eliminating irrelevant portions of the input before presenting it to a more focused object detector. The process of identifying irrelevant regions is driven by an efficient salience analysis of a deep CNN presented with the image. We have found that the selective attention mechanism of VDNets can substantially improve object detection performance. We have noted, however, that mistakes made by the selective attention system can be catastrophic, masking out image regions that contain objects of interest, making their detection impossible.

In this chapter, we propose a new method for extracting dense information about object location from pretrained networks, with the goal of improving selective attention in VDNets. Specifically, these new VDNets make use of channel-wise attention levels, as well as spatially specific attentional information from all receptive fields. By stacking these different kinds of attention maps, we can potentially preserve the semantic (object class) information that comes from late layers in the network while incorporating spatial location information that is richly represented in early layers. By using these stacked attention maps, we aim to improve the VDNet selective attention process and, therefore, produce better object detection performance.

The human visual attention system supports our naturally strong visual perception ca-

pabilities, so we see it as a useful guide for assessing selective attention behavior. Thus, in addition to measuring the object detection performance of our proposed system, we investigated possible relationships between information in VDNets and the patterns of attention exhibited by humans observing the same images. We used the distribution of fixation points produced by human subjects, measured using eye-tracking technology, as a measure of how the visual system distributes attention over images.

In overview, this chapter makes these contributions:

- Our approach uses a top-down salience analysis to identify irrelevant image regions, introducing a selective attention mechanism that masks out noise and unimportant background information in the image.
- Our measure of top-down salience involves a sensitivity analysis based on all filters and all spatial locations in order to generate a weighted attention map for every single layer in the selective attention network.
- We demonstrate the utility of dense salience maps, aggregated across layers, for preserving appropriate attentional information.
- We report the results of a human eye-tracking experiment, using the data to generate a kind of selective attention "ground-truth". (The human data are publicly available[1].) We compare computer vision selective attention mechanisms to human behavior.
- We show that our method performs favorably in comparison to state-of-the-art object detection approaches, demonstrating an improvement of up to 5% (mAP) in comparsion to the best of those approaches.

4.2 Algorithm

4.2.1 Dense Attention Ventral Networks

Our investigations into VDNets use a pretrained image classification network to guide selective attention. In our initial studies, described in the previous chapters, for a given

[1] `https://github.com/mkebrahimpour/Human_Attention_GT`

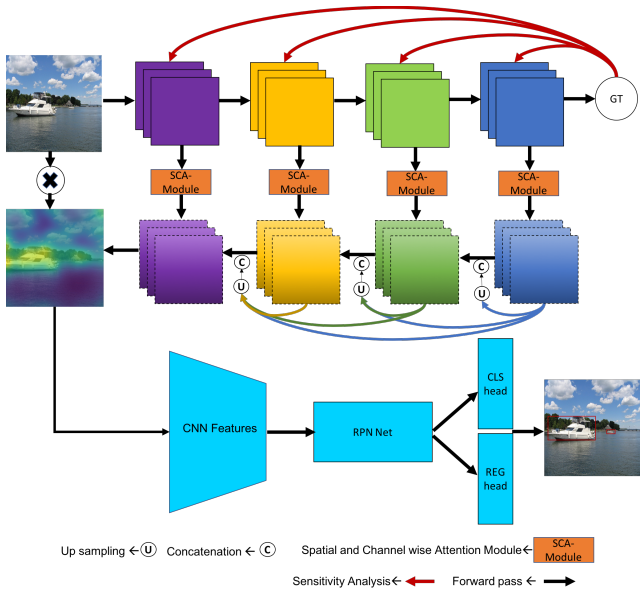

Figure 4.1: Dense VDNet: An object detection framework based on selective attention driven by densely stacked attention maps.

input image, the sensitivity of activity late in the network to each pixel was efficiently calculated, and regions containing low sensitivity pixels were masked out. We made two major modifications to this approach to produce the results reported in this chapter. First, we calculated the saliency of image regions based on activity at layers throughout the network, rather than only at the last convolutional layer. We aggregated spatial and channel-wise sensitivity information at each layer to produce layer-specific attention maps. Second, we densely stacked the whole collection of attention maps, from all of the layers, in order to drive selective attention to image regions. The goal was to improve the attention mechanism of VDNets to produce better object detection performance.

Spatial & Channel-wise Attention (SCA) Module

Spatial Attention. Generally speaking, objects occupy only parts of images, leaving background regions that can distract and misinform object detection systems. Instead of considering all parts of an image equally, spatial attention can focus processing on foreground regions, supporting the extraction of features most relevant for object class and object extent.

We represent a convolutional feature of layer n by $f_n \in \mathbb{R}^{W \times H \times C}$, where W and H

are the spatial dimensions of the rectangular layer and C is the number of feature channels in the layer. Spatial positions are specified by coordinate pairs: $\mathbb{L} = \{(x, y)|x = 1, 2, \ldots, W; y = 1, 2, \ldots, H\}$.

For a layer in a pretrained image classification network, the layer-specific spatial attention map is generated by the following equation:

$$A_n^s = W_n^s \odot f_n, \tag{4.1}$$

where W_n are weights that indicate the importance of each spatial location, across all of the convolutional filters. We initially calculate these weights based on the sensitivity of the Gestalt Total (GT) activation of the network to the feature. The Gestalt Total is calculated from the activation of the last convolutional layer, $A_{\textit{last}}$, as follows:

$$GT = \frac{1}{H \times W \times C} \sum_{i,j,k} A_{\textit{last}}(i, j, k) \tag{4.2}$$

The sensitivity of GT to a feature at layer n is the following:

$$\begin{aligned} G_n &= \frac{\partial GT}{\partial f_n} \\ W_n^u(x, y) &= \sum_c^C G_n(x, y, c), \end{aligned} \tag{4.3}$$

where W_n^u is not normalized (i.e., the weights are not in the $[0, 1]$ range). To normalize the weights for each location, ℓ, we apply a softmax operation to the weights spatially:

$$W_n^s(l) = \frac{\exp(W_n^u(l))}{\sum_{i \in \mathbb{L}} \exp(W_n^u(i))}, \tag{4.4}$$

where $W_n^s(l)$ denotes the weight for location l in layer n.

Channel-Wise Attention. The spatial attention calculation assigns weights to spatial locations, which addresses the problem of distractions from background regions. There is another way in which distractions can arise, however. Specific channels at a given layer can be distracting. When dealing with convolutional features, most existing methods treat all channels without distinction. However, different channels are often different in their relevance for objects of specific classes. Here, we introduce a channel-wise attention mechanism that assigns larger weights to channels to which the GT is sensitive, given the currently presented image. Incorporating these channel-wise attentional weights are intended to reduce this kind of distracting interference.

For channel-wise attention, we unfold f_n as$f = [f_n^1, f_n^2, \ldots, f_n^C]$, where $f_n^i \in \mathbb{R}^{W \times H}$ is the i$^{\text{th}}$ slice of f_n, and C is the total number of channels. The goal is to calculate a weight, W_n^c, to scale the convolutional features according to a channel-specific assessment of relevance:

$$A_n^c = W_n^c \cdot f_n, \tag{4.5}$$

Computing W_n^c is facilitated by the fact that we already have the sensitivities, G_n. Thus, an initial value for the weights can be had by setting $\hat{W}_n^c(c) = \sum_{x \in W, y \in H} G_n(x, y, c)$. These weights can be normalized to the $[0, 1]$ range using the softmax function:

$$W_n^c(c) = \frac{\exp(\hat{W}_n^c(c))}{\sum_{i \in C} \exp(\hat{W}_n^u(i))}, \tag{4.6}$$

These are the final channel-wise weights for layer n.

Dense Attention Maps

Given the spatial attention weights and the channel-wise attention weights, an attention weighted feature for layer n is calculated as:

$$f_n^{SCA} = A_n^c \cdot f_n + A_n^s \odot f_n, \tag{4.7}$$

with $f_n^{SCA} \in \mathbb{R}^{W \times H \times C}$. These weighted features are concatenated across layers, incrementally from the last layer to the first, but this is done after up-scaling lower spatial resolution layers. For the last layer, m, the map is simply the weighted features: $A_m = f_m^{SCA}$. For earlier layers:

$$\begin{aligned} A_i = [UP(f_m^{SCA}), UP(f_{m-1}^{SCA}), \ldots, f_i^{SCA}] \\ i = \{1, \ldots, m-1\} \end{aligned} \tag{4.8}$$

The result is a dense combination maps that is intended to incorporate both semantic information from the late layers and spatial information from the early layers. These attention maps then are combined to make a single map for the whole image. Since each layer can have a different number of channels, we simplify this aggregation by averaging each layer's attention map across channels, tranforming each A_i into a $W \times H$ matrix. The aggregated attention maps at each layer are computed as:

$$Att_i = A_m^{SCA} \oplus A_{m-1}^{SCA} \oplus \ldots \oplus A_i^{SCA}. \tag{4.9}$$

The final aggregated attention map, at the first layer, is Att_1. We smooth this $W \times H$ map by convolving it with a Gaussian filter, and then we threshold the result. This produces a binary mask specifying regions of relevance. The original image is multiplied by this mask to produce the input to the Dorsal Network.

It is important to note that the attention weights (W_n^s and W_n^c) are not learned as part of a training process. We begin with a pretained image classification network for the Ventral Network, and the attention weights are efficiently calculated for each layer when a given image is presented to that network.

Object Detection Network

Blanking irrelevant regions in the image substantially reduces the space of candidate regions to consider during object detection. In the Dorsal Network, the masked image is provided as input to a deep CNN trained to propose regions of interest with anchor boxes, process the contents of those regions, and output both class labels and bounding box coordinates. This is the approach used by Faster-RCNN [61]. The Dorsal Network is trained using a dataset of images that are annotated with both ground-truth class labels and ground-truth bounding boxes. As before, network parameters are selected so as to minimize a combination of the classification loss and the regression loss arising from the output of bounding box coordinates:

$$\begin{aligned} L(p_i, t_i) = & \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) \\ & + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*). \end{aligned} \tag{4.10}$$

where i is the index of an anchor box appearing in the current training mini-batch and p_i is the predicted probability of anchor i containing an object of interest. The ground-truth label p_i^* is 1 if anchor i is positive for object presence, and it is 0 otherwise. The predicted bounding box is captured by the 4 element vector t_i, and t_i^* contains the coordinates of the ground-truth bounding box associated with a positive anchor. The two components of the loss function are normalized by N_{cls} and N_{reg}, and they are weighted by a balancing parameter, λ. In our current implementation, the classification loss term is normalized by the mini-batch size (i.e., $N_{cls} = 32$) and the bounding box regression loss term is normalized

Table 4.1: PASCAL VOC 2007 Test Detection Results. Note that the minimum dimension of the input image for Faster and R-FCN is 600, and the speed is less than 10 frames per second. SSD300 indicates the input image dimension of SSD is 300×300. Large input sizes can lead to better results, but this increases running times. All models on the union of the trainval set from VOC 2007 and VOC 2012 and tested on the VOC 2007 test set, except for the model labeled Dense VDNet* which has been trained on the trainval set from VOC 2007 and tested on 2007 test set.

Method	Network	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster [61]	VGG	73.2	76.5	79	70.9	65.5	52.1	83.1	84.7	86.4	52	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83	72.6
ION [5]	VGG	75.6	79.2	83.1	77.6	65.6	54.9	85.4	85.1	87	54.4	80.6	73.8	85.3	82.2	82.2	74.4	47.1	75.8	72.7	84.2	80.4
Faster [36]	Residual-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72
MR-CNN [29]	VGG	78.2	80.3	84.1	78.5	70.8	68.5	88	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85	76.4	48.5	76.3	75.5	85	81
R-FCN [14]	Residual-101	80.5	79.9	87.2	81.5	72	69.8	86.8	88.5	89.8	67	88.1	74.5	89.8	90.6	79.9	81.2	53.7	81.8	81.5	85.9	79.9
SSD300 [47]	VGG	77.5	79.5	83.9	76	69.6	50.5	87	85.7	88.1	60.3	81.5	77	86.1	87.5	83.9	79.4	52.3	77.9	79.5	87.6	76.8
SSD512 [47]	VGG	79.5	84.8	85.1	81.5	73	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79	86.6	80
DSSD321 [28]	Residual-101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5	78.7	86.7	88.7	86.7	79.7	51.7	78	80.9	87.2	79.4
DSSD513 [28]	Residual-101	81.5	86.6	86.2	82.6	74.9	62.5	89	88.7	88.8	65.2	87	78.7	88.2	89	87.5	83.7	51.1	86.3	81.6	85.7	83.7
STDN300 [93]	DenseNet-169	78.1	81.1	86.9	76.4	69.2	52.4	87.7	84.2	88.3	60.2	81.3	77.6	86.6	88.9	87.8	76.8	51.8	78.4	81.3	87.5	77.8
STDN321 [93]	DenseNet-169	79.3	81.2	88.3	78.1	72.2	54.3	87.6	86.5	88.8	63.5	83.2	79.4	86.1	89.3	88.0	77.3	52.5	80.3	80.8	86.3	82.1
STDN513 [93]	DenseNet-169	80.9	86.1	89.3	79.5	74.3	61.9	88.5	88.3	89.4	67.4	86.5	79.5	86.4	89.2	88.5	79.3	53.0	77.9	81.4	86.6	85.5
Dense VDNet*	YOLONet	61.4	66.5	69.5	61.9	39.2	42.8	69.2.5	65.3	79.3	44.2	57.3	54.8	72.9	77.1	69.1	72.4	34.2	49.5	63.1	76.1	65.2
Dense VDNet	YOLONet	63.7	80.1	73.2	54.1	47.2	43.1	74.5	73.2	78.6	42.1	62.8	57.3	74.9	77.7	73.6	73.2	30.2	53.1	64.1	75.9	65.9
Dense VDNet	Resnet-101	**86.7**	95.8	98.4	98.2	66.4	94.6	90.2	96.1	71.2	72.8	54.9	97.9	95.6	89.6	90.3	93.2	69.6	89.2	82.1	93.2	74.1

Table 4.2: PASCAL VOC 2012 Test Detection Results. Note that the performance of Dense VDNet is about 4% better than baseline Faster-RCNN.

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
HyperNet-VGG [42]	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.97	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
HyperNet-SP [42]	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO [58]	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	89.4	49.4	75.5	57.0	87.5	80.9	81.0	74.7	41.8	71.5	68.5	82.1	67.2
MR-CNN-S-CNN [29]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [61]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
NoC [62]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Dense VDNet	**74.1**	85.7	82.2	74.1	55.9	60.9	79.8	76.4	83.5	56.9	78.4	63.4	88.4	83.9	81.3	77.1	46.5	74.6	65.3	80.1	72.9

by the number of anchor locations (i.e., $N_{reg} \approx 2,400$). As before, we set $\lambda = 10$, making the two loss terms roughly equally weighted (due to differences in scale).

It is worth noting that our general approach could easily incorporate other object detection algorithms for the Dorsal Network. The only requirement is that the object detection algorithm must be able to accept masked input images. For the results presented in this chapter, we have used a leading region proposal based approach, due to the high accuracy values reported for these methods in the literature. Having the Ventral Network reduce the number of proposed regions was expected to speed the object detection process and also potentially improve accuracy by removing from consideration irrelevant portions of the image.

Table 4.3: Comparison on COCO test-dev reveals that Dense VDNet performs favorably against state-of-the-art methods.

Method	Backbone	Input Resolution	AP	AP_{50}	AP_{75}
Faster R-CNN w/ FPN [45]	ResNet-101	1000×600	49.5	59.1	39.0
Deformable-CNN [13]	Inception-ResNet	1000×600	-	58.0	-
Deep Regionlets [83]	ResNet-101	1000×600	-	59.8	-
YOLOv2 [59]	DarkNet-19	544×544	31.6	44.0	19.2
YOLOv3 [60]	DarkNet-53	608×608	46.1	57.9	34.4
SSD [47]	ResNet-101	513×513	41.8	50.4	33.3
DSSD [28]	ResNet-101	513×513	44.2	53.3	35.2
RetinaNet [46]	ResNet-101	1333×800	50.7	59.1	42.3
Dense VDNet (ours)	Resnet 101	511×511	51.6	57.8	45.3

4.3 Experimental Results

4.3.1 Experiment Design and Implementation

We evaluated the Dense VDNet object detection model on PASCAL VOC 2007 [26] and PASCAL VOC 2012 [27]. We also compared the attention model to human performance.

The PASCAL VOC 2007 dataset has 20 classes and $9,963$ images which have been equally split into a training/validation set and a test set. The PASCAL VOC 2012 dataset contains $54,900$ images from 20 different categories, and it has been split approximately equally into a training/validation set and a test set. For PASCAL VOC 2007, we conducted training on the union of the VOC 2007 trainval set and the VOC 2012 trainval set, and we evaluated the results using the VOC 2007 test set. (This regimen is standard practice for these datasets.) For PASCAL VOC 2012, we performed training on its trainval set, and we evaluated the result on its test set. To evaluate performance, we used the standard mean average precision (mAP) measure. We report mAP scores using IoU thresholds at 0.5.

For networks with 224×224 image inputs, using PASCAL VOC, we trained the model with a mini-batch size of 1 due to GPU memory constraints. We started the learning rate at 3×10^{-4} for the first $900,000$ epochs. We then decreased it to 3×10^{-5} until epoch $1,200,000$. Then, we decreased it to 3×10^{-6} until epoch $2,000,000$. In all cases, we used

a momentum optimizer value of 0.9.

4.3.2 PASCAL VOC 2007 Results

The results of the PASCAL VOC 2007 dataset evaluation appear in Table 4.1. For the Ventral Network, we utilized VGG16 pretrained on the ImageNet dataset. We removed the fully connected layers and softmax calculation from VGG16, and we calculated GT based on the last convolutional layer. No fine tuning of parameters was done. For the Dorsal Network we used Resnet 101 pretrained on the ImageNet dataset as the backbone network. We also tried using YOLO V2 for object detection. The model was trained on 4 GPUs.

We compare our performance results with those reported for a variety of state-of-the-art approaches to object detection. Our primary baseline was Faster-RCNN using a Resnet 101 network trained on PASCAL VOC 2007. As shown in Table 4.1, the selective attention process of our approach (Dense VDNet) resulted in substantially better performance in comparison to Faster-RCNN and other methods. Dense VDNet appears to be more accurate at detecting larger objects than smaller ones, perhaps because the region proposal network based its output on the last (lowest resolution) convolutional layer. For most of the object classes, Dense VDNet performed better than other methods by a large margin. One shot detectors did not perform as well as the region proposal based approaches. We observed this by training YOLO V2 on PASCAL VOC 2007 and the union of PASCAL VOC 2007 and 2012 dataset and using the trained YOLO V2 network for the Dorsal Network. The difference likely reflects a trade off between speed and accuracy.

4.3.3 PASCAL VOC 2012 Results

We also measured Dense VDNet performance on the PASCAL VOC 2012 dataset. The Ventral Network was VGG16, with features for calculating GT extracted from the last convolutional layer. The Dorsal Network was initialized with parameters previously learned for the PASCAL VOC 2007 evaluation. For the additional training, the learning rate was initialized to 3×10^{-4} for $900,000$ epochs, and then it was reduced to 3×10^{-5} until reaching epoch $1,200,000$. The learning rate was further reduced to 3×10^{-6} until reaching $3,000,000$ epochs. The whole training process took about 20 days on 4 GPUs. This

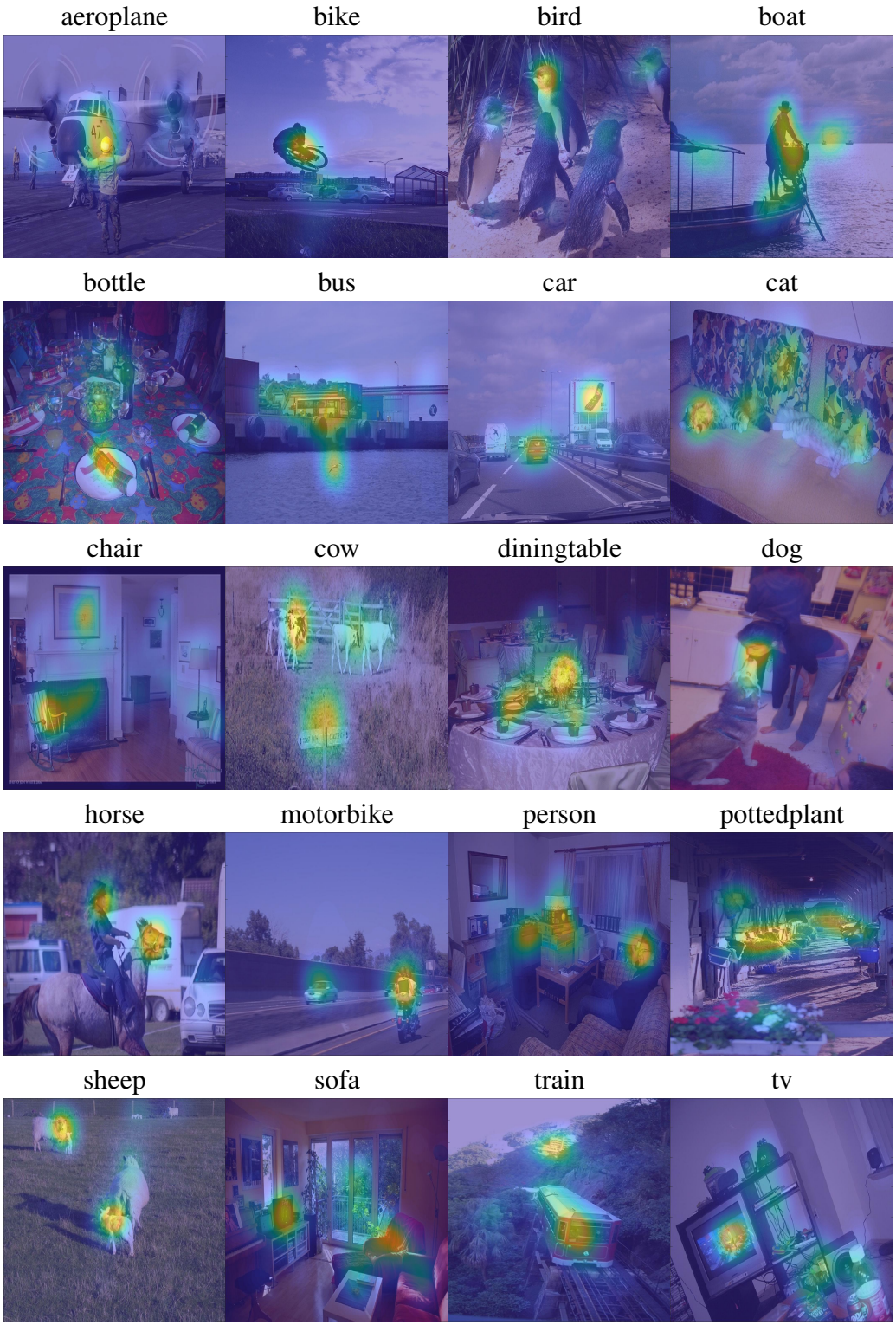

Figure 4.2: Example human eye fixation distributions on PASCAL VOC images.

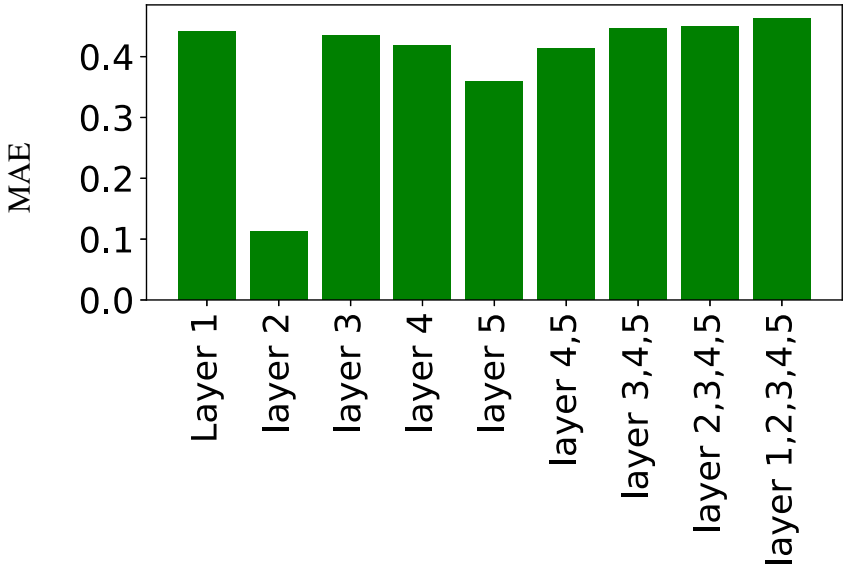

Figure 4.3: MAE measure between human fixation distributions and attention maps at different layers of Dense VDNet. The X-axis contains the layers of our Ventral Net. The names containing multiple numbers indicate the combination of these multiple layers with spatial and channel wise attention procedure. Our results indicate that the second layer in our neural network is the closest to human attention.

resulted in a Dense VDNet that produced comparable or better performance than state-of-the-art methods. Performance results for PASCAL VOC 2012 are shown in Table 4.2.

4.3.4 COCO Results

We also measured Dense VDNets performance on the COCO dataset. The Ventral Net was VGG16, with features for calculating GT extracted from the last convolutional layer. The Dorsal Net was Resnet 101. Initialized with parameters previously learned for the PASCAL VOC 2012 evaluation, the network was further trained using the COCO training-dev set. The training was done on 4 GPUs. This resulted in Dense VDNets that produced comparable or better performance than state-of-the-art methods, often by a large margin. Performance results for COCO are shown in Table 4.3.

4.3.5 Eye Tracking Study

Participants. 15 healthy undergraduate students (12 female,3 male; age: mean$\pm$s.d.$=$ 20.06 ± 1.62) were recruited from a University of California campus. Participants provided informed consent in accordance with IRB protocols and received one hour of course credit for their participation. Participation was restricted to those who reported normal, uncorrected vision in a pre-screen survey.

Materials. The images seen were a subset of 200 images from PASCAL VOC 2007. For display, images were scaled up to double their original size. The display width of images ranged from $636 - 1000$ pixels (mean$=$ 953.68; median$=$ 1000; standard deviation=102.54). The display height of images ranged from $350-1000$ pixels (mean$=$ 759.71; median$=$ 750; standard deviation$=$ 120.73). The ordering of images was randomized within participants.

Procedure. Participants completed the task individually in a research laboratory. Participants were seated at a desk in front of a computer and were fitted with a head-mounted Eyelink II eye tracker system. A microphone was placed nearby to record participants' speech. Prior to the beginning of the experiment, the eye-tracker was calibrated using a standard nine-point grid, and the subject was shown how to perform a drift correction, which took place at the beginning of each trial. Eye movement data was collected via the Eyelink control software and custom MATLAB scripts. Data from the right eye were collected using both pupil shape and corneal reflection.

Each trial began with the participant fixating the center of the screen and pressing the space bar to initiate the trial. Then, an image was displayed in the center of the screen for 5 seconds. Participants were instructed to name out loud as many different objects as they could identify in the image, within the 5-second time limit.

Fixation Heat-Map. The raw eye-tracking data were converted into MatLab data structures using the `Edf2Mat` package. Heat maps were generated from the eye-fixation data. Fixations were included in the analysis only if they began at or after the start of the trial. Fixations were pooled from all participants. For each image, a zero matrix with the same $N \times M$ dimensions as the original image pixel height and width were created. Fixation coordinates were scaled so that they corresponded to locations within the original image sizes and then rounded to the nearest integer. Thus, the coordinate values for each fixation

corresponded to a location within the matrix. For each fixation, the value of the corresponding matrix position was increased by the duration of the fixation in milliseconds. Then, all values of the matrix were divided by the maximum value of the matrix in order to normalize all matrix values to the $[0, 1]$ range. Finally, a convolution was performed on the matrix using a Gaussian kernel ($\sigma = 20$, size = 80×80). These steps yielded a heat map showing the likely places towards which participants attend within the images. Calculations were performed in MATLAB using custom scripts.

4.3.6 Eye-Tracking Study Results

Example fixation distribution heatmaps are provided in Figure 4.3. We used these distributions as a form of ground-truth for analyzing the attention maps produced by Dense VDNet. We compared the attention map distributions, $G(x, y)$, with the human fixation distributions, $S(x, y)$, using a simple "MAE" measure of average absolute difference:

$$MAE = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} |S(x, y) - G(x, y)| \tag{4.11}$$

where W and H are the width and height of the image. We performed this assessment for various attention maps in the network. Some of these comparison measures are shown in Figure 5.9.

None of the network layers produced attention maps that were very good fits to human fixation behavior. Eye movements and fixations likely change the vision problem in fundamental ways. Still we found that the second convolutional layer displayed the greatest similarity to human performance. This contrasts with the attention mechanisms in other object detection frameworks, which frequently base attention only on the last convolutional layer. It appears as if the high resolution and fundamental features learned by the network provide good guides for attention during object detection.

4.4 Summary

In this chapter, we highlighted the utility of incorporating selective attention mechanisms into object detection algorithms. We suggested that such mechanisms could guide the search over image regions, focusing this search in an informed manner. In addition,

we demonstrated that the resulting removal of distracting irrelevant material can improve object detection accuracy substantially. Importantly, we aggregated sensitivity information across scales.

Our approach was inspired by the visual system of the human brain. Theories of spatial attention that see it as arising from dual interacting "what" and "where" visual streams led us to propose a dual network architecture for object detection. The resulting architecture, Dense VDNet, integrates attention based object detection methods with supervised approaches.

The benefits of selective attention, as implemented in Dense VDNet, are evident in the performance results reported on the PASCAL VOC 2007 and PASCAL VOC 2012 datasets. Evaluation experiments revealed that Dense VDNet displays greater object detection accuracy than state-of-the-art approaches, often by a large margin.

Finally, it is worth noting, once again, that Dense VDNet has a very general and flexible structure. The Ventral Network and the Dorsal Network can easily be updated to incorporate the latest advancements in image classification and object detection algorithms. In this way, we see Dense VDNet as an approach with many potential instantiation.

Chapter 5

Deep Neural Network Attention vs. Human Attention

5.1 Introduction

The impressive capabilities of CNNs, which have met or even surpassed human performance in some vision tasks [49], are somewhat surprising given how little is actually understood about their internal operations. Due to the large number of nonlinear interactions inside these networks, ANNs have long been treated as "black boxes", with their inner workings opaque to even their creators. However, just as the "black box" perspective on the human mind gave way to the development of new methods and theoretical tools, researchers in computer science have recently begun finding new ways of understanding the intermediate representations produced by ANNs [87]. For example, methods of "deep visualization" examine the representations learned by individual artificial neurons by iteratively generating a synthetic input image that maximally activates each neuron. The images produced in this process are often surprisingly interpretable by the human eye [49], suggesting that these networks may learn feature representations that are perhaps similar to those of the human visual system.

Of course, ANNs have have been at the heart of powerful models of human cognitive processes for over three decades [65]. In the field of computational cognitive neuroscience, ANNs have served a crucial role in discriminating between various proposed models of the structure and dynamics of cognitive and brain systems [50]. By comparing the perfor-

mance of human participants to ANNs of various designs, researchers can investigate the feasibility of specific network models for explaining aspects of brain and behavior. For example, while most CNNs designed for object recognition are purely feedforward, Rajaei and colleagues recently argued, based on neuroimaging data and computational models, that recurrent connections are crucial for performance under cases of degraded input, such as partial object occlusion [55]. This serves to illustrate that, by examining where humans succeed and ANNs fail, we can improve both our understanding of the brain and create more effective computational models.

In this spirit, computer scientists have been taking inspiration from the human visual system to improve the speed and accuracy of CNNs for object detection. In particular, visual selective attention has been proposed as one mechanism that is crucial for human object detection performance, but such a mechanism is absent from most algorithms designed for the same purpose. Several CNN techniques have now been proposed that implement some form of selective attention [92, 69, 23]. Indeed, it has been the focus of the previous chapters. For most of these techniques it is unknown how these attention algorithms compare to human selective attention. Examining the differences and similarities between human overt visual attention and the attentional representations produced in these CNNs will help to hone our understanding of specifically which features of the human visual system may most fruitfully be modeled by CNNs (and for which tasks), or, indeed, whether deep networks ofter a promising approach to understanding human vision, at all. Given the relative power of human vision in comparison to state-of-the-art object detection CNNs, comparisons of this kind might also suggest new and better biologically-inspired approaches to selective attention in computer vision.

For the sake of comparison, we took human behavior to be approximately normative with regard to the allocation of attention. As reported in previous chapter, we recorded the eye motions of humans as they detected objects in still images, and we took these data as indicating the image regions most worthy of overt selective attention. We then examined a variety of CNN approaches to attention, including the *Densely Connected Attention Model* (Dense VD Nets) presented in the previous chapter. We assessed the approaches with regard to their correspondance to human performance. The resulting analyses have produced insights into both human selective attention and the design of computer vision object detection systems. By identifying the layers within the Densely Connected Atten-

tion Model that best capture human selective attention, we were able to identify the sort of visual features that best predict the distribution of human eye fixations. By comparing various CNN methods, we were able to characterize their relative strengths and weaknesses for attention-guided object detection.

5.2 Eye Tracking Study

In this chapter, we make use of the same eye tracking data that were introduced in the last chapter. For ease of reference, the method used to collect those data are reported, here.
Participants. Our participants were 15 healthy undergraduate students (12 female, 3 male; age: mean$\pm$s.d.= 20.06 $\pm$1.62) at one of the University of California campuses. Participants received one hour of course credit for their participation. Participants provided informed consent in accordance with IRB protocols. Participation was restricted to individuals with normal vision, as reported on a pre-screen survey.
Materials. The stimuli were a subset of 200 images drawn from the PASCAL Visual Object Classes 2007 database [26]. Each image was scaled up to twice its original size for display on a 1920x1080 screen. The display width of images was between 636-1000 pixels, subtending 23.83-34.76 degrees of visual angle. The display height of images was between 350-1000 pixels, subtending 13.68-34.76 degrees of visual angle. Images were centered on a black background.
Procedure. Participants completed an object detection task individually in the lab. Participants were seated at a desk in front of a computer screen and wore a head-mounted Eyelink II eye tracker. The eye tracker was set to pupil/corneal reflection recording mode and sampled at a rate of 250 Hz. A microphone was placed near participants to record their speech. Before beginning the task, the eye tracker was calibrated with a nine-point grid. Participants were also shown how to perform a drift correction, which occurred prior to each trial. Eye movement data was collected with the Eyelink software and custom Matlab scripts, implemented with the Psychophysics toolbox Matlab package [7].

Each trial started with a drift correction in which participants had to fixate a center dot and press the space bar to initiate the trial. Participants were allowed to pause on the drift correction screen for as long as they wished before initiating a trial. Then, a randomly selected image was displayed on the screen for 5000 ms. Participants were instructed to

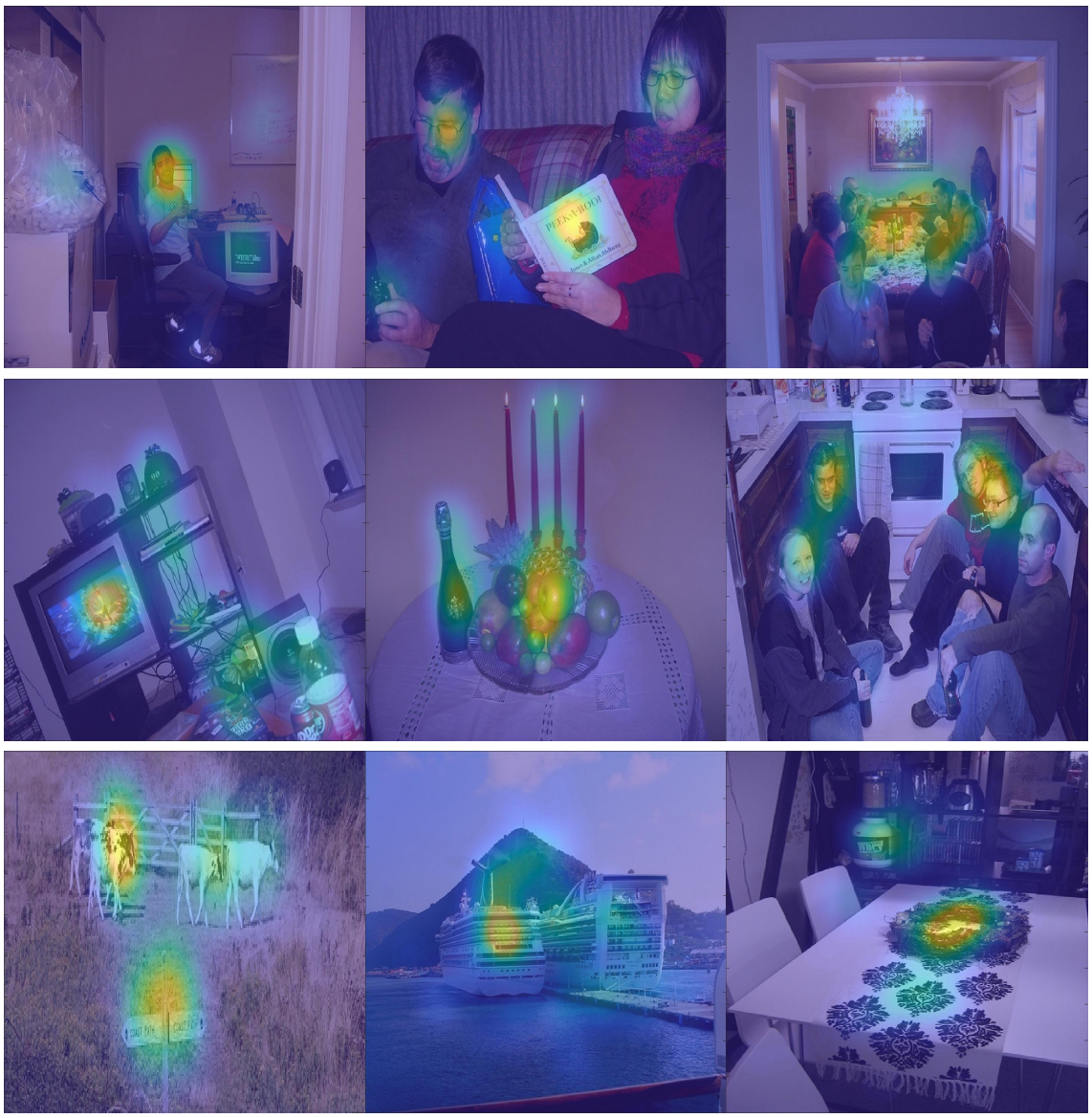

Figure 5.1: Results of the human eye tracking study on some of our test images. The heatmaps reveal the distribution of attention.

name out loud as many unique objects as they could detect within the time limit. This was repeated until participants had viewed all 200 images, once each. The experiment took about 30 minutes to complete, with $\sim$10 minutes dedicated to setup and calibration and $\sim$20 spent on the task itself.

Data Processing. The raw eye tracking data was first converted into a matlab-compatible data structure using the `Edf2Mat` package. Then, custom scripts were used to generate fixation heatmaps for each image. Recorded fixations contributed to the heatmaps only if they fell entirely within the period of display. Fixations were first pooled from all participants. Then, for each image a zero matrix was generated with the same dimensions as the pixel dimensions of the original (before scaling for display) image. Fixation coordinates from the display images were then scaled down to map onto the coordinates of the original sizes. Each possible fixation position then corresponded to a position within the zero matrix for that image. For each recorded fixation at a given location, the value of the corresponding cell in the zero matrix was increased by the duration of the fixation in milliseconds. Then, all values were divided by the maximum value in the matrix such that possible values ranged between zero and one. Finally, we performed a convolution over the matrix with a Gaussian kernel ($\sigma = 20$, size = 80×80). This process generated a fixation heatmap showing the relative likelihood of fixations occurring at each region of each image. Examples of the human eye tracking overt visual attention maps are illustrated in Figure 5.1.

5.3 Attention Models

In recent years, deep CNNs have demonstrated human level performance on image classification tasks when given large amounts of supervised training data. This impressive performance has led to a growing interest in understanding the internal representations learned by the networks [25, 89, 87, 72]. Of particular interest, here, are the mechanisms of "selective attention" in these networks that allow them to focus on relevant portions of input images. In this section, we review several previously published approaches and then review our novel CNN architecture: the *Densely Connected Attention Model*.

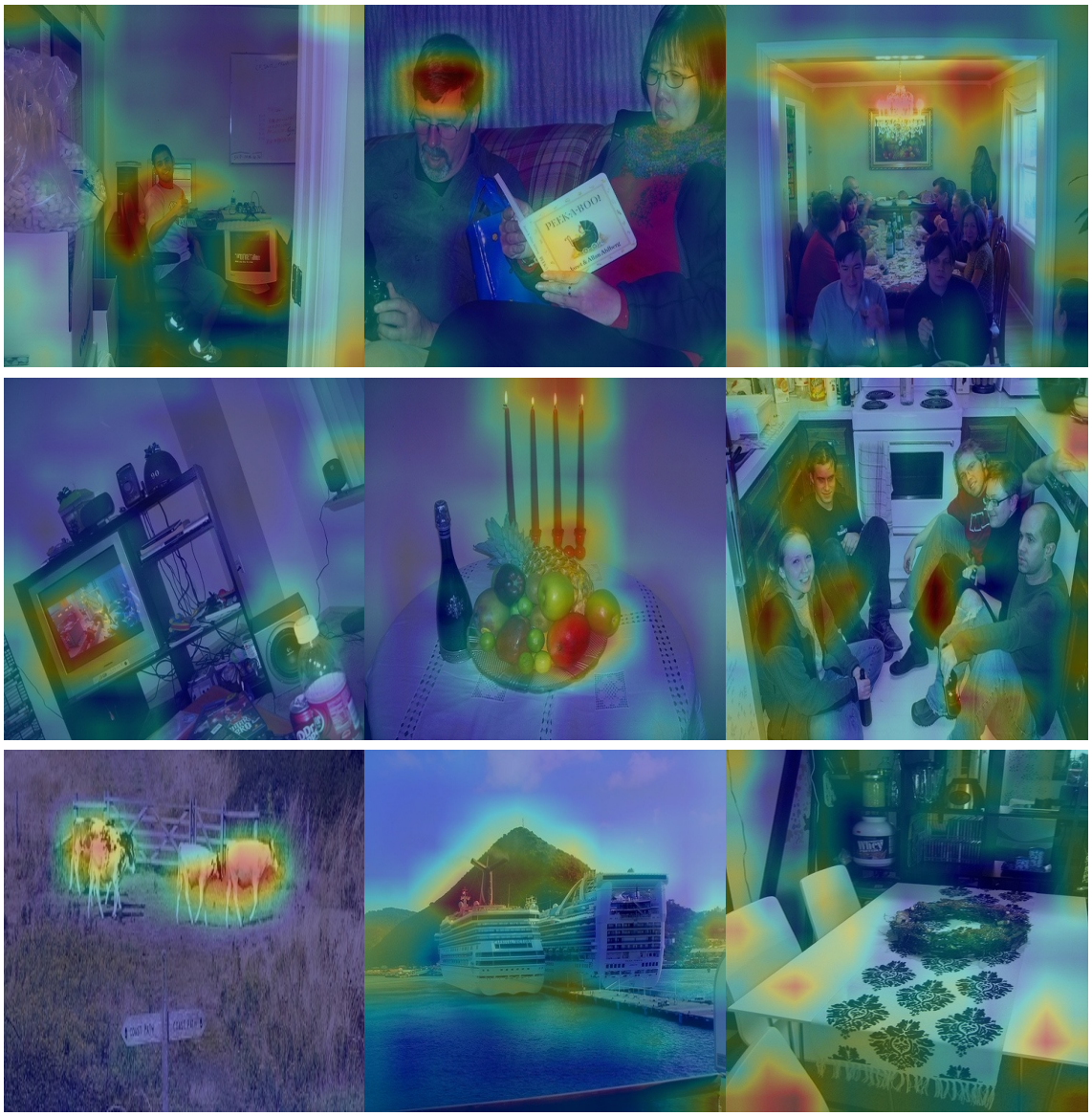

Figure 5.2: Results of the CAM attention algorithm. The heatmaps reveal the attention of the CNN.

5.3.1 Class Activation Maps

Zhou et al. proposed a technique called *Class Activation Maps (CAM)* [92]. These were predicated on the observation that deep CNN architectures for computer vision lose spatial information close to the network output due to the use of fully connected layers after the convolutional layers. The authors proposed "chopping off" the fully connected layers and calculating the *Global Average Pooling (GAP)* of activation at the last convolutional layer, aggregating the spatial information in each channel. For a given image, let $f_k(x, y)$ represent the activation of filter k in the last convolutional layer at spatial location (x, y). Then, for filter k, the result of performing GAP is $F^k = \sum_{x,y} f_k(x, y)$. A linear regression can be conducted between the F^k values and each output score, S_c, for each object class, c, producing regression weights, w_k^c. Given these weights, the CAM for a given image, with regard to object class, c, is:

$$M_c(x, y) = \sum_k w_k^c f_k(x, y) \tag{5.1}$$

The authors argued that using this approach would preserve more spatial location information for the main objects in the image. Examples of CAMs are depicted in Figure 5.2. This method has a number of drawbacks. Specifically, determining the regression weights takes time, and using the attention information in the suggested manner can noticeably decrease classification performance.

5.3.2 Gradient Based Class Activation Maps

With the goal of improving on the CAM method, Selvaraju and colleagues proposed a different approach [69]. They argued that the weights for each channel that are used in CAMs are implicit in the CNN, itself, so there is no need to perform additional regressions [69]. They suggested taking the derivative of the winning (i.e., greatest) object class output with regard to the activation in the last convolutional layer of the network:

$$A = \frac{\partial y_c}{\partial f} \tag{5.2}$$

where y_c is the output activation for object class, c, and $f \in \mathbb{R}^{W \times H \times K}$ is the last convolutional layer. The derivative values are aggregated channel-wise into a matrix, A, and the

weights for each channel are computed as:

$$a_k^c = \frac{1}{W \times H} \sum_{x \in W} \sum_{y \in H} A(x, y) \tag{5.3}$$

This weight, a_k^c, is intended to capture the "importance" of channel k for a target class c. The corresponding attention maps are calculated similarly to CAMs, though values are rectified (ReLU) to be positive:

$$L_{Grad-CAM}^c = ReLU(\sum_k a_k^c f_k) \tag{5.4}$$

Notice that this results in a coarse heatmap of the same size as the convolutional feature maps (14×14 in the case of the last convolutional layers of the VGG network). Since this method made use of gradient information, it is called *Grad-CAM*. Example Grad-CAM attention maps are illustrated in Figure 5.3.

5.3.3 Guided-Backpropagation

The *Guided-Backpropagation* technique was proposed as a way to train CNNs so as to make activation in convolutional feature maps easier to interpret [76]. The goal was to encourage units to only become active if they are contributing to the activation of the appropriate object class output. This would make the spatial location of active units indicative of where objects were. The method involved modifying the standard *backpropagation of error* algorithm [65] by surpressing negative gradients. Unit error (delta) values were rectified, hindering the formation of strong negative connection weights. The bias toward positive weights made the "meaning" of learned features easier to interpret. A very simple example of this learning process is illustrated in Figure 5.4. Figure 5.5 shows the resulting attention maps (where activation is high) for some example images.

5.3.4 Guided-Gradient Based Class Activation Maps

In order to obtain information about both relevant image regions and relevant pixels within regions, Selvaraju and colleagues fused Guided-Backpropagation with Gradient Based Class Activation Maps, producing the *Guided Grad-CAM* method [69]. The merging of the previous algorithms was done in a simple manner. The attention maps resulting

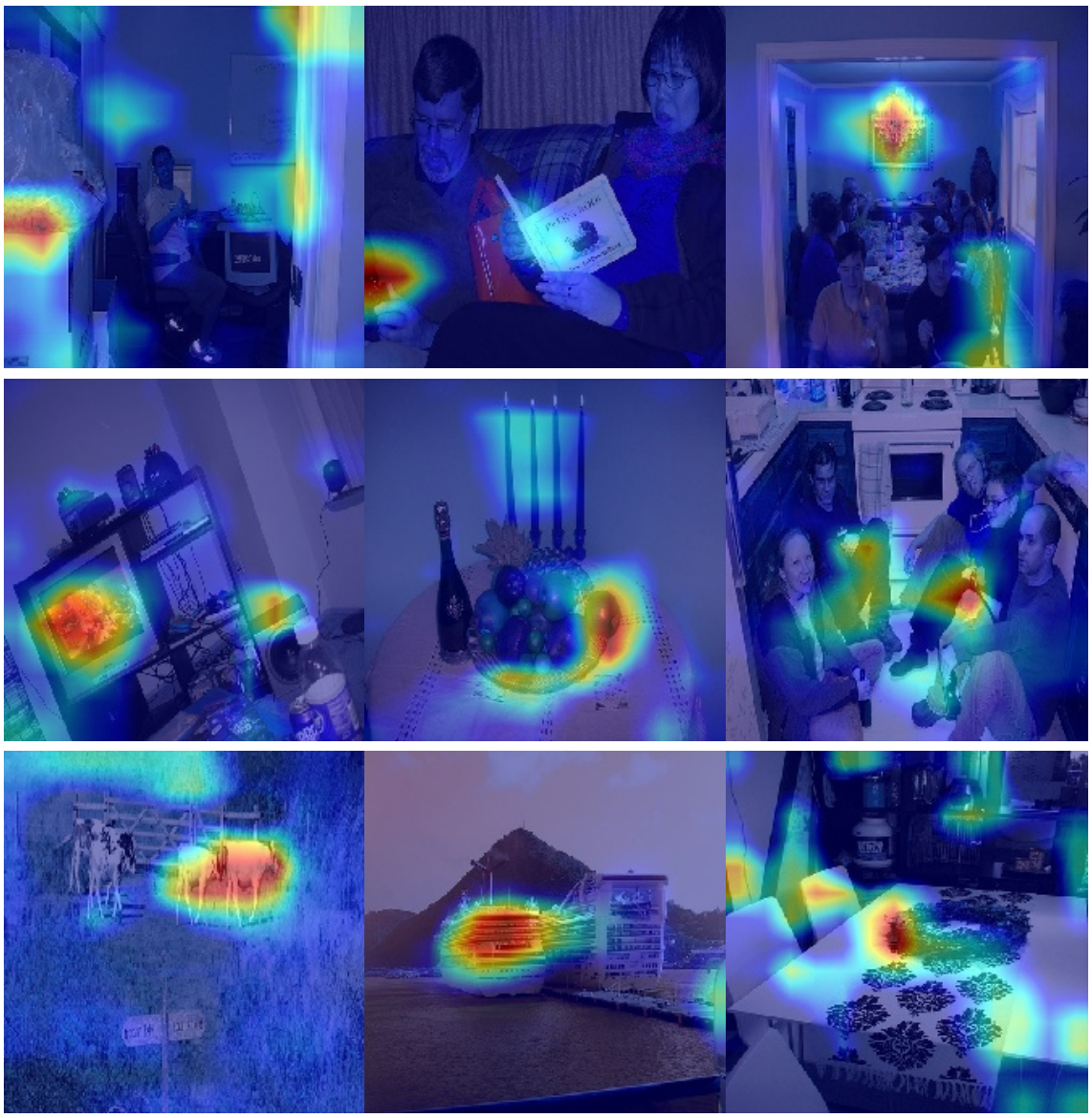

Figure 5.3: Results of the Grad-CAM attention algorithm. The heatmaps show attended regions.

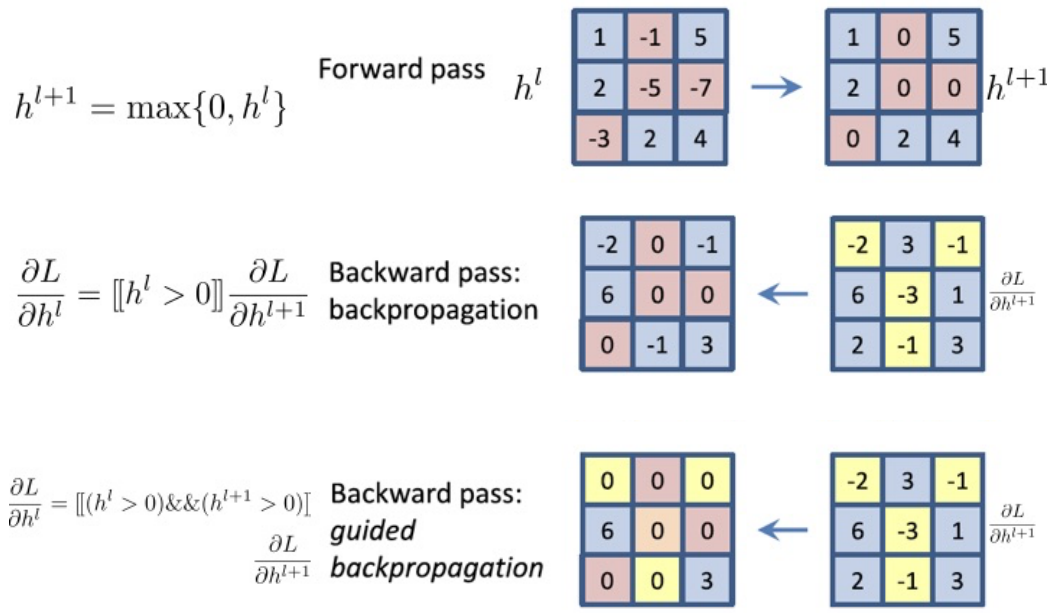

Figure 5.4: An illustration of the training process used in guided-backpropagation.

from the two algorithms were combined using point-wise multiplication. Some example results are displayed in Figure 5.6.

5.3.5 The Densely Connected Attention Model

All of these CNN attentional mechanisms rely on activation in the last convolutional layer. Information in earlier layers of the networks is ignored. It is possible, however, that useful guidance for attention might be found in the earlier convolutional layers, where spatial resolution tends to be higher and detected visual features tend to be more simple and smaller. Our proposed *Densely Connected Attention Model* (Dense VDNets) assembles information from each channel and each spatial location across all of the layers in the CNN image classifier. Our method is fast and efficient, and it does not require any additional training of the CNN.

Spatial Attention. Generally speaking, objects occupy only portions of images, leaving background regions that can distract and misinform object detection systems. Instead of considering all parts of an image equally, spatial attention can focus processing on foreground regions, supporting the extraction of features most relevant for determining object class and object extent. Here, we review our proposed approach, which was introduced in

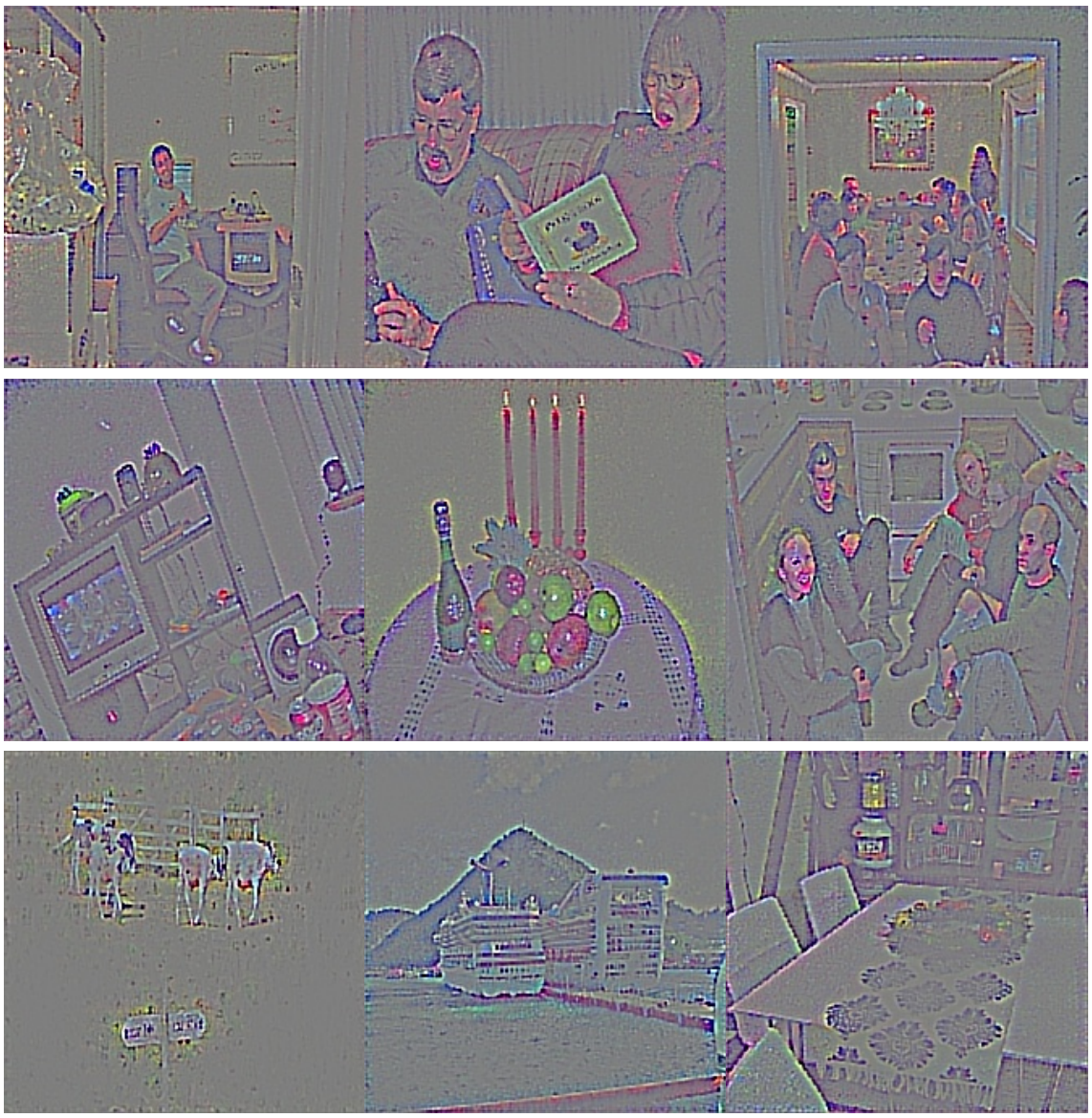

Figure 5.5: Results of the Guided-Backpropagation attention algorithm. The color-coded maps show attended pixels.

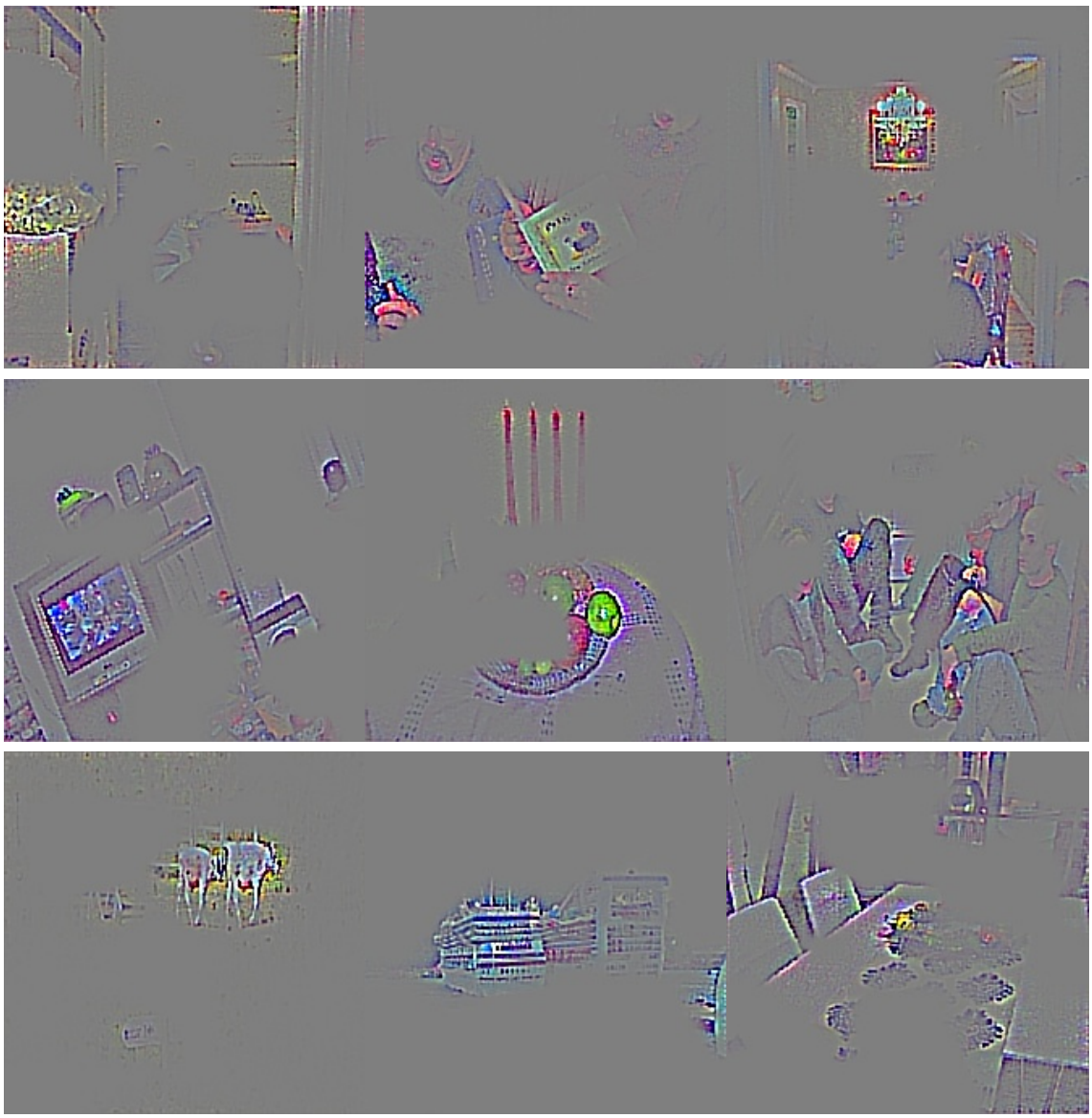

Figure 5.6: Results of the Guided-Grad-CAM attention algorithm. The color-coded maps show attended pixels.

the previous chapter. A schematic of our approach is shown in Figure 5.7.

Formally, the annotation that we use to represent the activation of convolutional feature layer n is $f_n \in \mathbb{R}^{W \times H \times C}$, where W and H are the spatial dimensions of the rectangular layer and C is the number of feature channels in the layer. Spatial positions are specified by coordinate pairs: $\mathbb{L} = \{(x, y)|x = 1, 2, \ldots, W; y = 1, 2, \ldots, H\}$.

For layer n in a pretrained image classification network, the layer-specific spatial attention map is ...

$$A_n^s = W_n^s \odot f_n \tag{5.5}$$

... where W_n^s are weights that indicate the importance of each spatial location, across all of the convolutional channels. We initially calculate these weights based on the sensitivity of the *Gestalt Total (GT)* activation of the network to the feature [23]. The Gestalt Total is calculated from the activation of the last convolutional layer, A_{last}, as follows:

$$GT = \frac{1}{H \times W \times C} \sum_{i,j,k} A_{last}^s(i, j, k) \tag{5.6}$$

The sensitivity of GT to a feature at layer n is ...

$$G_n = \frac{\partial GT}{\partial f_n} \tag{5.7}$$

$$\hat{W}_n^s(x, y) = \textstyle\sum_c^C G_n(x, y, c), \tag{5.8}$$

... where $\hat{W}_n^s$ is not normalized (i.e., the weights are not in the $[0, 1]$ range). To normalize the weights for each location, we apply a softmax operation to the weights spatially ...

$$W_n^s(x, y) = \frac{\exp(\hat{W}_n^s(x, y))}{\sum_{i \in W, j \in H)} \exp(\hat{W}_n^s(i, j))}, \tag{5.9}$$

... where $W_n^s(x, y)$ denotes the weight for location (x, y) in layer n.

Channel-Wise Attention. The spatial attention calculation assigns weights to spatial locations, which addresses the problem of distractions from background regions. There is another way in which distractions can arise, however. Specific channels at a given layer can be distracting. When dealing with convolutional features, most of the existing methods treat all channels without distinction. However, different channels often have different degrees of relevance for objects of specific classes. Here, we introduce a channel-wise attention mechanism that assigns larger weights to channels for which the GT is sensitive,

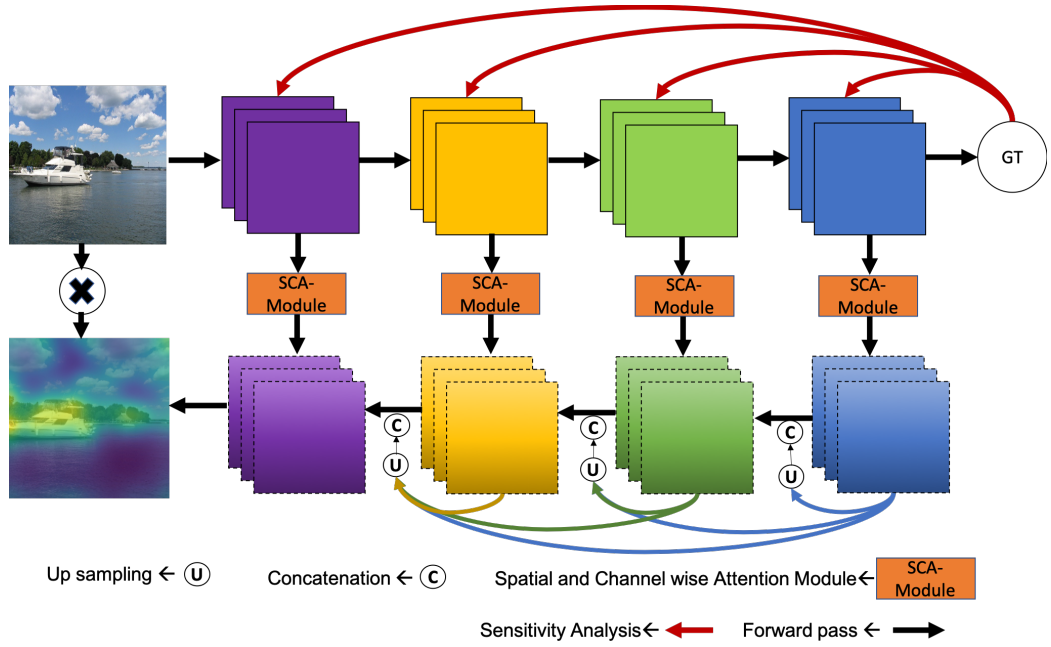

Figure 5.7: The network architecture of the *Densely Connected Attention Model*.

given the currently presented image. Incorporating these channel-wise attentional weights are intended to reduce this kind of distracting interference.

For channel-wise attention, we unfold f_n as$f = [f_n^1, f_n^2, \ldots, f_n^C]$, where $f_n^i \in \mathbb{R}^{W \times H}$ is the i$^{\text{th}}$ channel slice of f_n, and C is the total number of channels. The goal is to calculate a weight, W_n^c, to scale features according to a channel-specific assessment of relevance, allowing for the construction of a layer-specific channel-wise attention map:

$$A_n^c = W_n^c \cdot f_n \tag{5.10}$$

Computing W_n^c is facilitated by the fact that we already have the sensitivities, G_n. Thus, an initial value for the weights can be found by setting $\hat{W}_n^c(c) = \sum_{x \in W, y \in H} G_n(x, y, c)$. These weights can be normalized to the $[0, 1]$ range using the softmax function:

$$W_n^c(c) = \frac{\exp(\hat{W}_n^c(c))}{\sum_{i \in C} \exp(\hat{W}_n^c(i))}, \tag{5.11}$$

These are the final channel-wise weights for layer n.

Dense Attention Maps. Given the spatial attention weights and the channel-wise attention weights, an attention weighted feature for layer n is calculated as ...

$$f_n^{SCA} = A_n^c \cdot f_n + A_n^s \odot f_n, \tag{5.12}$$

...with $f_n^{SCA} \in \mathbb{R}^{W \times H \times C}$. (In Figure 5.7, this computation is done in the *Spatial and Channel wise Attention Module – SCA-Module*.) For the last convolutional layer, m, the attention map is simply the weighted features: $A_m = f_m^{SCA}$. For earlier layers, the weighted features are concatenated across layers, incrementally from the last layer to the first, but this is done after up-scaling lower spatial resolution (later) convolutional layers:

$$A_i^{SCA} = [UP(f_m^{SCA}), UP(f_{m-1}^{SCA}), \ldots, f_i^{SCA}] \quad i = \{1, \ldots, m-1\} \tag{5.13}$$

This process generates dense combination maps that are intended to incorporate both semantic information from the late layers and higher resolution spatial information from the early layers. The maps are aggregated to produce a single attention map for the whole image. Since each layer can have a different number of channels, we simplify this aggregation by averaging each layer's attention map across channels, transforming each A_i^{SCA} into a $W \times H$ matrix. The aggregated attention maps at each layer are then computed as:

$$A_i = A_m^{SCA} + A_{m-1}^{SCA} + \ldots + A_i^{SCA}. \tag{5.14}$$

It is important to note that the attention weights (W_n^s and W_n^c) are not learned as part of a training process. We begin with a pretained image classification network (VGG16 [73]), and the attention weights are efficiently calculated for each layer when a given image is presented to that network. Some resulting attention maps from our model are shown in Figure 5.8.

5.4 Comparing CNN and Human Attention Maps

We have reviewed several existing CNN attention algorithms as well as our novel Densely Connected Attention Model which incorporates spatial attention as well as channel-wise attention in each layer. We compared the attention maps of the various CNNs with overt visual attention maps produced from our eye tracking data, using exactly the same images in all cases. Our summary comparison statistic was the mean absolute error (MAE) between attention maps ...

$$MAE = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} |S(x,y) - G(x,y)| \tag{5.15}$$

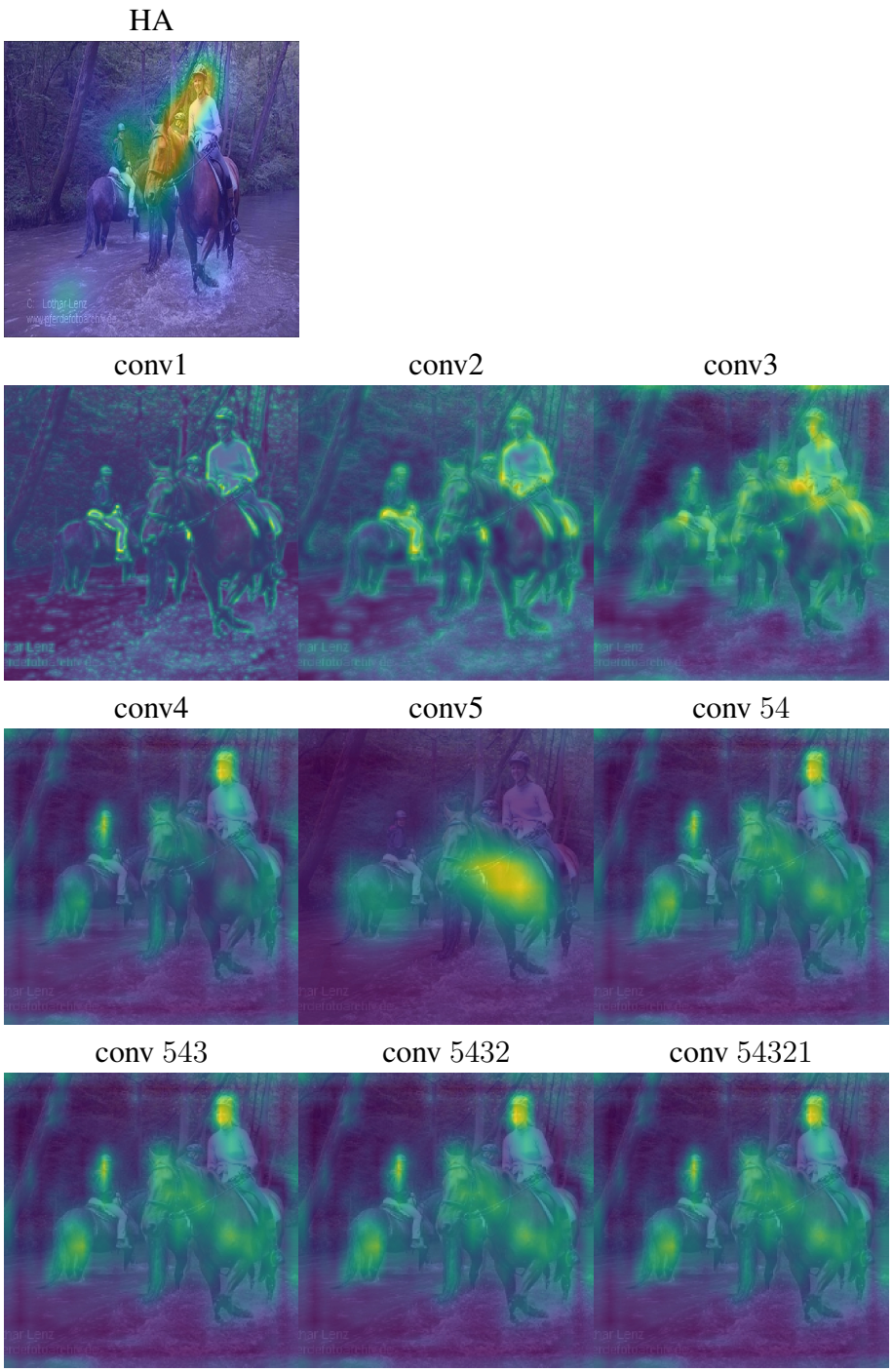

Figure 5.8: Results of the Densely Connected Attention Model for each convolutional layer, as well as for combinations of layers. The numbers following the "conv" label indicate layer number. For instance, "conv 54" refers to the combination of layers 5 and 4. Human eye tracking results for this example image are also shown (HA).

... where W and H are the width and height of the image, $S(.)$ is the attention map from a network, and $G(.)$ is the "ground truth" (human performance) attention map. We compared different CNN attention methods, but we also examined the fit of various attention maps at individual internal layers of the Densely Connected Attention Model. The resulting error values are summarized in Figure 5.9.

These results reveal that the CAM algorithm provides the closest match to overt visual human attention on our test images, while the second layer in our proposed attention mechanism comes in second place. The fact that layer 2 of our network is much closer to the attentional patterns of human participants than later layers is quite surprising, given the total reliance on the final convolutional layer seen in other mechanisms. However, while the combination of layers in our attention mechanism makes the attentional pattern quite dissimilar to that of humans, it also produces improved performance in object detection tasks, as described in the previous chapter, allegedly due to the combination of semantic object class information in later layers with early, more spatially precise, lower-level visual feature information.

5.5 Summary

We investigated various deep CNN visual attention mechanisms and compared their attentional patterns to that of human participants. We also revisited our novel attention algorithm which integrates more kinds of information, at multiple scales, than the other approaches. We found that the CAM algorithm provides the best match to human performance, with the second convolutional layer of our Densely Connected Attention Model ranking second. In general, none of the CNN algorithms provided a paricularly good match to overt visual human attention, however. Thus, while deep CNNs may learn a hierarchy of visual features similar to the response properties of neurons in the human visual system [84], current attentional mechanisms in CNNs do not seem to align with human overt attention.

This suggests that human attention may not be a good guide for improving the object detection performance of deep CNNs. We have found that the attention maps produced by CAM, Grad-CAM, Guided-Backpropagation, and Guided Grad-CAM tend to focus on a single salient object in the image. In contrast, the Densely Connected Attention Model ap-

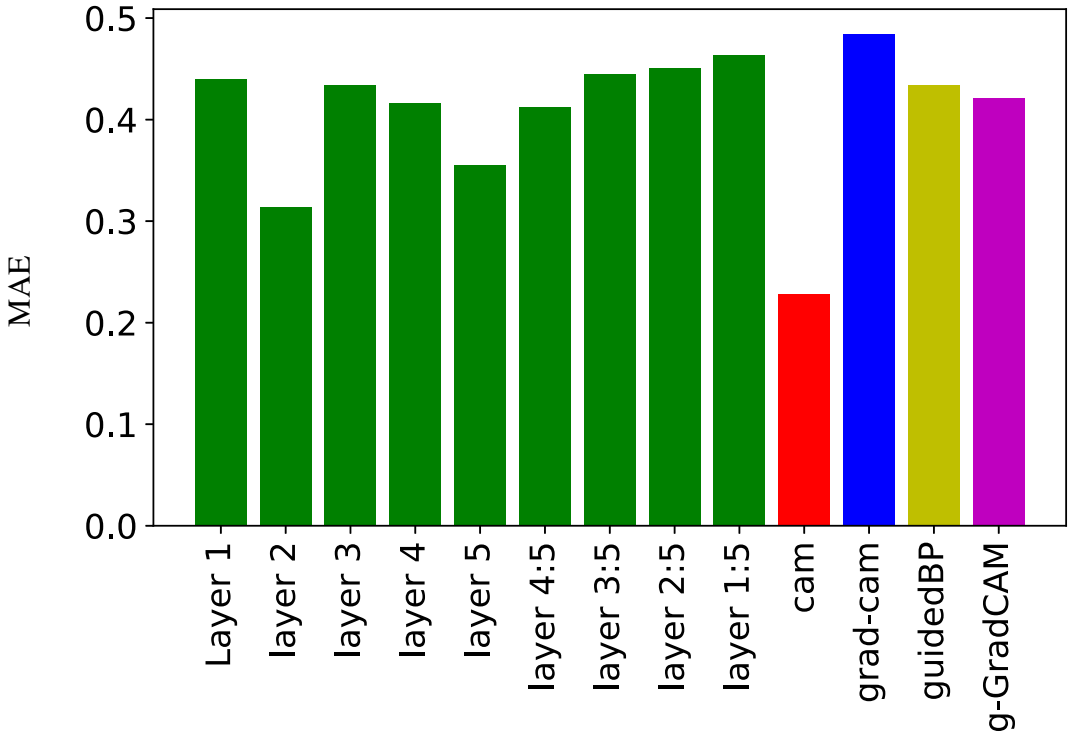

Figure 5.9: Error between human attention maps and those produced at various layers of the Densely Connected Attention Model. Also shown are errors for other algorithms.

pears to attend to all objects in the image, ignoring background distractions. Despite these benefits, the CNN attention maps in our model were quite different than those of humans, with the greatest similarity appearing in layers that encode fairly low-level features. We are left with interesting questions concerning the nature of the differences between CNN object detection and human vision that give rise to this mismatch of attentional patterns.

Chapter 6

End-to-End Auditory Object Recognition via Inception Nucleus

6.1 Introduction

One of the advantages of deep CNNs in object recognition is their ability to learn useful features in an end-to-end manner by mapping raw data, such as RGB pixels, to class labels. In contrast, auditory object recognition is typically implemented based on engineered features [10, 77]. One of the most powerful types of engineered representation for speech recognition tasks is based on the mel-frequency cepstrum [52], which is basically the discrete cosine transform of the windowed spectra. Researchers have used such engineered features as inputs to CNNs for audio classification tasks, such as Automatic Speech Recognition (ASR) [66] and music analysis [82]. In these cases, CNNs are typically applied to two-dimensional feature maps created by arranging the log-mel cepstral features of each frame along the time axis. This feature map creates locality in both time and frequency domains [2], which means that the machine learning problem can be framed as an image classification problem.

However, cepstral features were designed specifically for speech recognition and may not be optimal for other types of audio classification tasks. More generally, pre-engineered features will be tailored to whatever the problem is to be solved, which means they may not be readily transferred to other problem domains. Another potential problem with engineered features is that they must be computed as inputs to the classification system, such

as a deep learning convolutional network. On-line computation of spectral or cepstral features can be costly in terms of time and power, especially for edge computing applications that do not have access to cloud computing servers. More recently, researchers have developed deep learning networks that take raw waveforms as input, rather than using pre-engineered features. This approach is known as end-to-end audio classification. For instance, Dai et al. proposed five CNNs with different architectures and a varying number of parameters [15]. They achieved impressive accuracy on the Urbansound8k dataset [15]. Tokozume and Harada proposed EnvNet which is an 8-layer neural network that takes the raw waveform as input, but requires careful selection of hyperparameters to choose appropriately sized kernels [79]. AclNet [40] is another end-to-end CNN architecture, inspired by MobileNet [37] because of its computational efficiency. AclNet achieved human-level accuracy for the ESC50 dataset with only 155k parameters and 49.3 million multiply-adds per second [40]. Finally, Ravanelli and Benjio proposed speaker recognition network based on raw wavforms [56].

We present a deep CNN that learns to classify broad categories of sounds directly from raw audio waveforms. In comparison to previous end-to-end audio classification efforts [15, 79, 40], we make use of a novel combination of 1D and 2D convolutional layers, and, most importantly, "inception nucleus" layers. The inception nucleus approach, described in Section 6.2, reduces sensitivity to prespecified filter sizes by depending on adaptation during learning. In comparison to prior work, the proposed method also greatly reduces the number of parameters while outperforming current state-of-the-art networks on the UrbanSound8k dataset by 10.4 percentage points. Thus, our CNN is a strong candidate for low-power always-on sound classification applications. In addition, we analyze the learned representations, using visualizations to reveal wavelet-like transforms in early layers, supporting deeper representations that are discriminative and meaningful, even with reduced dimensionality.

6.2 Proposed Method

Our proposed end-to-end neural network takes time-domain waveform data — not engineered representations — and processes it through several 1D convolutions, the inception nucleus, and 2D convolutions to map the input to the desired outputs. The details of the

Table 6.1: Our proposed deep neural networks architectures. Each column belongs to a network. The third row indicated number of parameters. Through out our experiments we designed different network architectures for the ablation study. The model to analyze the kernel sizes in the inception nucleus layer is called Filter Analysis (FA) model. Also there is a model to analyze Full Inception (FI) architecture. Also, to analyze the impact of Batch Normalization (BN), we have an architecture that implements BN in their layer. The convolutional layer parameters are denoted as "conv (1D or 2D),(number of channels),(kernel size),(stride)." Layers with batch normalization are denoted with BN.

Inception Nucleus Nets Configurations			
Inception	Inception-FA	Inception-FI	Inception-BN
289 K	789 K	479 K	292 K
Input (32000×1)			
Conv1D,32,80,4		Inception Nucleus: Conv1D,32,60,4 Conv1D,[32,80,4]×2 Conv1D,[32,100,4]×2	Conv1D,32,80,4 with BN
Inception Nucleus: Conv1D,64,4,4 Conv1D,[64,8,4]×2 Conv1D,[64,16,4]×2	Inception Nucleus: Conv1D,64,20,4 Conv1D,[64,40,4]×2 Conv1D,[64,60,4]×2	Inception Nucleus: Conv1D,64,4,4 Conv1D,[64,8,4]×2 Conv1D,[64,16,4]×2	Inception Nucleus: Conv1D,64,4,4 - BN Conv1D,[64,8,4]×2-BN Conv1D,[64,16,4]×2-BN
Max Pooling 1D, 64,10,1			
Reshape (put the channels first)			
Conv2D,32,3×3,1			Conv2D,32,3×3,-BN
Max Pooling 2D,32,2×2,2			
Conv2D,64,3×3,1 Conv2D,64,3×3,1			Conv2D,64,3×3,1-BN Conv2D,64,3×3,1-BN
Max Pooling 2D,64,2×2,2			
Conv2D,128,3×3,1			Conv2D,128,3×3,1-BN
Max Pooling 2D,128,2×2,2			
Conv2D,10,1×1,1			Conv2D,10,1×1,1-BN
Global Average Pooling			
Softmax			

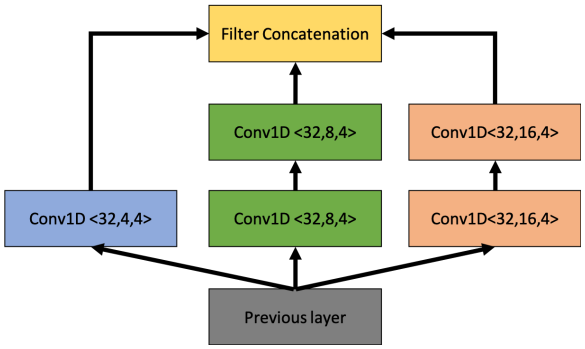

Figure 6.1: Inception nucleus. The input comes from the previous layer and is passed to the 1D convolutional layers with kernel sizes of 4, 8, and 16 to capture a variety of features. The convolutional layer parameters are denoted as "conv1D,(number of channels),(kernel size),(stride)." All of the receptive fields are concatenated channel-wise in the concatenation layer.

proposed architectures are described in Table 6.1. The overall design can be summarized as follows:

Fully Convolution Network. We propose an inception nucleus convolution layer that contains a series of 1D convolutional layers followed by nonlinearities (i.e., ReLU layer) to reduce the sensitivity of the architecture to kernel size. Convolutional networks are well-suited for audio signals for the following reasons. First, similar to images, we desire our network to be translation invariant with regard to time. Second, convolutional networks allow us to stack layers, which gives us the opportunity to detect higher-level features through a series of lower-level detectors. We used Global Average Pooling (GAP) in our architectures to aggregate the spatial information in the last convolutional layer and map this information onto class labels. GAP greatly reduces the number of parameters to make the network relatively light to implement.

Variable Length Input/Output. Since sound can vary in temporal length, we want our network to handle variable-length inputs. To do this, we use a fully convolutional network. The input layer to our network is a 1D array, representing the audio waveform, which is denoted as $\mathbf{X} \in \mathbb{R}^{32000 \times 1}$, since the audio files are about 4 seconds, and the sampling rate was set to be 8 *kHz*. The network is designed to learn a set of parameters, ω, to map the

input to the prediction, $\hat{Y}$, based on nested mapping functions, given by Eq 6.1.

$$\hat{Y} = F(\mathbf{X}|\omega) = f_k(...f_2(f1(\mathbf{X}|\omega_1)|\omega_2)|...\omega_k) \tag{6.1}$$

where k is the number of hidden layers and f_i is a typical convolution layer followed by a pooling operation.

Inception Nucleus Layer. We propose the use of an inception nucleus to produce a more robust architecture for sound classification. This approach also makes the architecture less sensitive to idiosyncratic variance in audio files. A schematic representation of the inception nucleus appears in Fig 6.1. The inputs to the inception nucleus are the feature maps of the previous layer. Then, three 1D convolutions with different kernels are applied to the inputs to capture a variety of features. We test the following kernel sizes in our experiments: $4, 8, 16, 20, 40, 60, 80, 100$. (See Section 6.3.) After obtaining the feature maps from our convolutional layers, we concatenate the receptive fields in a channel-wise manner.

Reshape. After applying 1D convolutions on the waveforms and obtaining low-level features, the feature map, L, will be $\in \mathbb{R}^{1\times m\times n}$. We can treat L as a grayscale image with width=m, height=n, and channel=1. For simplicity, we transpose the tensor L to $L' \in \mathbb{R}^{m\times n\times 1}$. From here, we apply normal 2D convolutions with the VGG standard kernel size of 3×3 and stride = 1 [73]. Also, the pooling layers have kernel sizes = 2×2 and stride = 2. We also implemented the inception nucleus with batch normalization to analyze the effect of batch normalization on our approach, as explained in Section 7.4.

Global Average Pooling (GAP). In the last convolutional layer we compute GAP to aggregate the most abstract features over the spatial dimensions and reduce the number of outputs to class labels. We use GAP instead of max pooling to reduce the number of parameters and avoid adding fully connected layers at the end of the network. It has been noted in the computer vision literature that aggregating features across spatial locations and channels keeps important information while reducing the number of parameters [23, 92]. We intentionally did not use fully connected layers with a softmax activation function to avoid overfitting, since fully connected layers greatly increase the number of parameters. GAP was implemented as follows:

$$GAP_c = \frac{1}{w \times h} \sum_{i,j} A(i, j, c) \tag{6.2}$$

where w, h, c are width, height, and channel of the last feature map (A).

6.3 Experimental Results

We tested our network on the UrbanSound8k dataset which contains 10 kinds of environmental sounds in urban areas, such as drilling, car horn, and children playing [67]. The dataset consists of $8,732$ audio clips of 4 seconds or less, totalling 9.7 hours. We padded zeros to the samples that were less than 4 seconds. To speed computation, the audio waveforms were down-sampled to 8 *kHz* and standardized to zero mean and unit variance. We shuffled the training data to enhance variability in the training set.

We trained the CNN models using the Adam [41] optimizer, a variant of stochastic gradient descent that adaptively tunes the step size for each dimension. We used glorot weight initialization [33] and trained each model with batch size 32 for up to 300 epochs until convergence.

To avoid overfitting, all weight parameters were penalized by their ℓ_2 norm, using a λ coefficient of 0.0001. Our models were implemented in Keras [12] and trained using a GeForce GTX 1080 Ti GPU.

For ablation study, we evaluated our proposed architecture in a variety of scenarios, Inception-BN indicates the model with Batch normalization to understand the impact of batch normalization on our architecture. Inception-FA shows the model with filter analysis, and Inception-FI indicate the fully inception model. Table 6.2 provides classification performance on the testing set along with numbers of parameters used for the Urbansound8k dataset. The table shows that our CNN outperformed other methods in terms of test classification accuracy, with the fewest number of parameters. Preliminary simulations revealed that fully connected layers at the end of the network caused overfitting due to an explosion in the number of weight parameters. These preliminary results led us to use a fully convolutional network with a reduced number of parameters.

We note that the deeper networks (M5, M11, and M18) can improve performance if their architectures are well-designed. However, our inception nucleus model is 88.4% accurate, which outperforms the reported test accuracy of CNNs on spectrogram input using the same dataset by a large margin [52]. Also, inception nucleus-FI achieves very good results in terms of both accuracy and number of parameters. This result suggests that if we let the network learn useful features for the desired task in the convolutional layers, recognition performance and generalization is improved over pre-engineered features.

Table 6.2: Accuracy of different approaches on the UrbanSound8k dataset. The first column indicates the name of the method, the second column is the accuracy of the model on the test set, the third column reveals the number of parameters. It is clear that our proposed method has the fewest number of parameters and achieves the highest test accuracy. Please note that Inception-BN indicates the model with Batch normalization, Inception-FA notes the model with filter analysis, and Inception-FI indicate the fully inception model.

Model	Test	# Parameters
M3-fc [15]	46.82%	129M
M5-fc [15]	62.76%	18M
M11-fc [15]	68.29%	1.8M
M18-fc [15]	64.93%	8.7M
M3-Big [15]	57.55%	0.5M
RCNN [68]	71.68%	3.7M
ACLNet [40]	65.32%	2M
EnvNet-v2 [79]	78%	101M
PiczakCNN [52]	73%	26M
VGG [53]	70%	77M
Inception Nucleus-BN (Ours)	**83.2%**	**292K**
Inception Nucleus-FA (Ours)	**70.9%**	**789K**
Inception Nucleus-FI (Ours)	**75.3%**	**479K**
Inception Nucleus (Ours)	**88.4%**	**289K**

Kernel Analysis. We also analyzed the learned kernels of our Inception Nucleus model in the very first layer of our neural network. Interestingly, the network learns wavelet transforms at the first convolutional layer, as has been found by other researchers [4, 80]. Some of those filters are illustrated in Fig 6.2.

Representation Analysis. To better understand the learned representations in the Inception Nucleus model, we extracted features from the last convolutional layer (before applying GAP) and applied T- distributed Stochastic Neighbor Embedding (t-SNE). t-SNE is a nonlinear dimensionality reduction technique for embedding high-dimensional data for visualization in a low-dimensional space of two or three dimensions [48]. Therefore, we applied t-SNE to reduce the dimensionality to two. The results, shown in Fig. 6.3, suggest that the network learned meaningful and discriminative features, as the different classes are fairly well distinguished from each other.

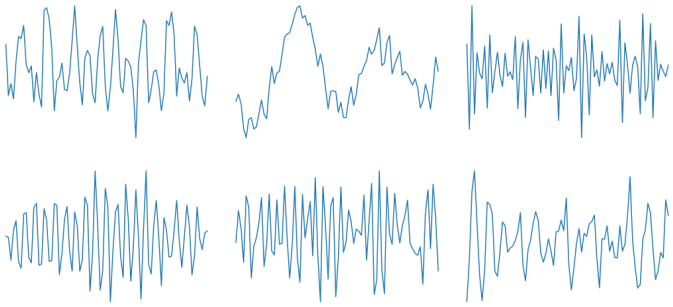

Figure 6.2: Illustrating 3 filters in the first convolutional layer. The visualization indicates that learned representations in the early layers implemented wavelet-like audio filters.

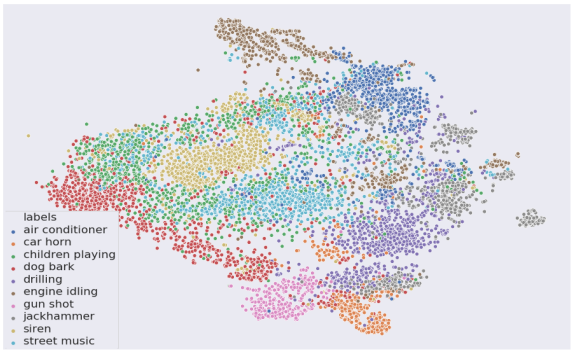

Figure 6.3: Illustration of the top two components of the t-SNE of the last convolutional layer.

Depth Analysis. We found that deeper networks with larger numbers of parameters were less generalizable as indicated by poorer performance on the test set. For example, M18 has $8.7M$ parameters (see Table 6.2) but only achieves 64.93% accuracy, compared with our inception nucleus network which achieves 88.4% by only having $289K$ parameters. This finding runs counter to results from the image recognition literature, in which deeper networks tend to perform better than shallower ones [36, 39, 38]. The observed detriment of additional hidden layers may be attributable to the limited number of training examples, which can be tested in future studies with larger datasets.

Kernel Size Analysis. Dai et al. [15] found that smaller kernel sizes are insufficient to capture the necessary bandpass filter characteristics in the earlier convolutional layers. Our results indicate that, with the Inception Nucleus-FA, large kernel sizes (e.g. $60, 80, 100$) are more effective in the first convolutional layer. By contrast, large kernel sizes in the

second layer reduce performance substantially (e.g., using the Inception Nuclus-FA with large kernels in the second layer decreased performance by 13 points). We conclude that a larger inception nucleus is more suitable for the first layer, with smaller kernels in later convolutional layers.

Batch Normalization. We tested whether batch normalization (BN) improves performance in our CNN, as it can for very deep neural networks. Without BN, our inception nucleus achieves 88.4% accuracy while with BN it achieves 83.2%. The slight decrease in accuracy using BN may have been observed because our CNNs did not have enough layers to show the advantage of BN.

6.4 Summary

In this study, we developed, optimized, and tested CNNs up to 13 layers deep that used an inception nucleus to overcome problems with choosing kernel sizes. The CNNs were trained to perform end-to-end sound classification, and they were benchmarked against the Urbansound8K dataset. Results from our networks, compared with competitors, showed better performance with fewer parameters — up to 88.4% accuracy using only $289K$ parameters. The ability to perform end-to-end computations effectively using so few parameters may be useful for edge computing applications, especially with optimized hardware, such as neuromorphic implementations of deep networks [6]. Our results indicate that end-to-end computation does not detract from performance by forgoing cepstral or spectral features. To the contrary, our networks outperformed competitors that used log-mel spectrogram inputs [52]. Visualizations of kernels learned in the earliest layer revealed wavelet-like transforms that build up to more abstract and discriminating learned features in deeper layers. In summary, we have demonstrated effective end-to-end sound classification with an efficient deep learning network.

Chapter 7

End-to-end Auditory Object Recognition of Infant Sounds

7.1 Introduction

Infants produce cries and other non-speech vocalizations from birth, and they start to produce more complex, speech-like vocalizations as early as 3 months of age [8, 51]. They enter a period of exploring speech-like vocalizations, known as babbling, and they may start speaking their first words around one year of age. But even before then, vocalizations carry information about physical and emotional states that matter to caregivers, as well as to infant health and well-being [8, 51]. Changes in vocalizations over time carry information about speech development that may be valuable for informing theories and diagnosing when development is atypical.

Studies of infant vocalizations often borrow analysis techniques formulated for speech analysis, such as mel-frequency cepstral coefficents [10, 77, 52]. These techniques rely on an engineered feature space based in part on theories of adult human speech perception and production. One of the most powerful types of engineered representation for speech recognition tasks is based on the mel-frequency cepstrum [52], which is basically the discrete cosine transform of the windowed spectra. Researchers have used such engineered features as inputs to machine learning models for automatic speech recognition (ASR) [66] and music classification [82]. In these cases, inputs are typically two-dimensional feature maps created by arranging the log-mel cepstral features of each frame along the time axis.

This feature map creates locality in both time and frequency domains [2], which means that the machine learning problem can be framed as an image classification problem.

With respect to classifying infant vocalizations, Rosita and Junaedi developed a system using MFCC features based on voice type [63]. They classified vocalizations into categories of hungry, discomfort, and tiredness [63], and found MFCC features are well suited for discriminating these categories. In another study, Alaie and colleagues used Gausian Mixture Models to classify healthy and sick infants based on infant cries using MFCC features with promising results [3]. Similarly, Zabidi et al. proposed a multi-layer perceptron to classify infant cries with Asphyxia [88].

While models have progressed to date based on traditional speech analysis techniques, it is worth noting that pre-linguistic vocalizations are often atypical relative to adult speech. More generally, acoustic-based features that are tailored to infant vocalizations and distinctions may be better suited for classification tasks and research on speech development. In recent years, advances in deep neural networks have given rise to so-called ”end-to-end” models that take mostly unprocessed audio and video signals as inputs, and learn layers of features that represent the signals along dimensions that are helpful for any given classification task.

In particular, convolutional neural networks (CNNs) have proven effective in learning to classify large sets of categories when given very large numbers of training examples [73, 36]. One of the advantages of deep CNNs in sound classification is their ability to learn useful features in an end-to-end manner by mapping raw data, such as raw waveform audio, onto class labels. For instance, Dai et al. proposed 5 CNNs with different architectures and a varying number of parameters [15]. AclNet [40] is another end-to-end CNN architecture, inspired by MobileNet [37] because of its computational efficiency. AclNet achieved human-level accuracy for the ESC50 dataset with only 155k parameters and 49.3 million multiply-adds per second [40].

In the present study, we take a data-driven approach to classifying infant vocalizations by applying an end-to-end deep learning model of sound classification to a database of infant and adult vocalizations [20]. Vocalizations recorded in natural settings served as inputs in the form of raw waveforms to convolutional network layers that learned representations useful for classifying vocalizations. We also tested whether an ”Inception Nucleus” layer might improve performance by providing more varied convolutional filters. This and other

features of our model architecture were designed to minimize the number of parameters and thereby avoid over-fitting to training data.

7.2 Infant Vocalization Dataset

In the present study, we trained the sound classification model introduced in the last chapter [20] on a database of sound recordings collected by Warlaumont and colleagues [54]. They recruited families with infants to participate in a study of infant speech development. Each family agreed to put a vest on their infant that had a pocket containing an audio recording device (LENA; [30]). The infant wore the vest for 12-hour periods at 3, 6, 9, and 18 months of age. Parents were advised to record during the weekend when they were most likely to be together with the infant, and the parents were compensated for their participation ($20 at 3-months, $30 at 6-months, $40 at 9-months, $60 at 18-months, and an additional $40 and a summary of the output from their recordings at the end of the study). On recording days, families otherwise went about their business as usual, and recordings captured all sounds near the infant, including their vocalizations, nearby adult vocalizations, and various other ambient sounds.

Three 5-minute periods of relatively active vocalizations by the infant were identified for each day-long recording with assistance of the LENA system. Research assistants transcribed each 5-minute period to identify time segments that contained infant and adult vocalizations. Infant vocalizations were categorized as laugh/cry (sometimes both in the same vocalization period), canonical babbling (well-structured syllables with consonants and vowels), and non-canonical babbling (lone vowels or consonants, less speech-like sounds). Adult vocalizations were categorized as either directed at the infant (IDS), such as vocal play meant to elicit responses, or directed at other adults or other children nearby (ADS). Any sensitive information (e.g. credit card numbers) that happened to be recorded was removed.

The dataset consisted of recordings from 15 different families. We did not have transcriptions for all four time periods for each family, so we had less data in the older age ranges. The distribution of data across different vocalizations categories and different ages is shown in Fig 7.1.

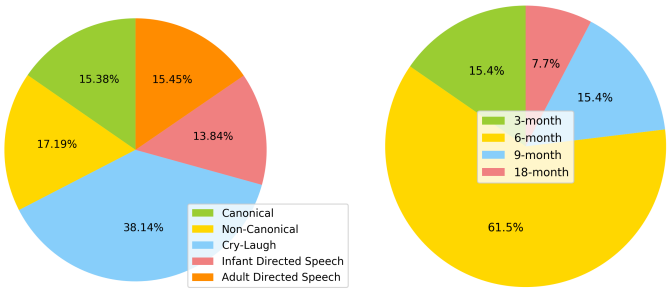

Figure 7.1: Left: The distribution data in different classes in InfantSound dataset. Right: the distribution of samples per age of infants.

7.3 Sound Classification Networks

We investigated two end-to-end neural networks that take time-domain waveform data as inputs and process them through several 1D and 2D convolutions to map onto the desired output classes. The main difference between the two networks was the inclusion or exclusion of an inception nucleus. The model architectures are diagrammed in in Table 7.1, and the main parts of both models are summarzied below, including the inception nucleus.

Convolution Layers. Convolutional networks are designed to learn filters that are passed over spatial or temporal dimensions, such as images or acoustic waveforms. Convolutional filters help models learn translation-invariant features, and they are efficient in terms of the number of parameters that need to be trained. Convolutional layers can also be stacked to learn representations that build on each other, from small, low-level features to larger, high-level objects. Finally, convolutional layers can handle inputs of varying lengths and features of varying scales.

The input layer to our network is a 1D array, representing the audio waveform, which is denoted as $X \in \mathbb{R}^{8000\times1}$, since the audio files are 1 second, and the sampling rate was set to be 8 *kHz*. The network is designed to learn a set of parameters, ω, to map the input to the prediction, $\hat{Y}$, based on nested mapping functions, given by Eq 7.1.

$$\hat{Y} = F(X|\omega) = f_k(...f_2(f1(X|\omega_1)|\omega_2)|...\omega_k) \tag{7.1}$$

where k is the number of hidden layers and f_i is a typical convolution layer followed by a pooling operation.

Inception Nucleus Layer. We use an inception nucleus [20] for more robust classi-

Table 7.1: Specific architectures of two different CNNs examined in the present study, with or without an Inception Nucleus layer. The convolutional layer parameters are denoted as "conv (1D or 2D),(number of channels),(kernel size),(stride)".

Neural Networks Configurations	
CNN with Inception	CNN without Inception
302 K	540 K
Input (8000×1)	
Conv1D,32,80,4	-
-	Conv1D,32,9,4
-	Conv1D,32,8,4
-	Conv1D,32,9,4
Inception Nucleus: Conv1D,64,4,4 Conv1D,[64,8,4]$\times$2 Conv1D,[64,16,4]$\times$2	-
Max Pooling 1D,10,1	Max Pooling 1D,2,1
Reshape (put the channels first)	
Conv2D,32,3×3,1	Conv2D,64,3×3,1
Max Pooling 2D,2×2,2	
Conv2D,64,3×3,1 Conv2D,64,3×3,1	Conv2D,128,3×3,1 Conv2D,128,3×3,1
Max Pooling 2D,2×2,2	
Conv2D,128,3×3,1	Conv2D,256,3×3,1
Max Pooling 2D, 2×2,2	
Conv2D,10,3×3,1 -	Conv2D,10,1×1,1 Conv2D,10,1×1,1
Flattening	
Softmax	

fication and reduced sensitivity to idiosyncratic variance in audio files. The inputs to the inception nucleus are the feature maps of the previous layer, and the nucleus itself consisted of three 1D convolutions with different kernels that are simultaneously applied to inputs in order to capture a range of features and scales. Kernel sizes have significant impacts on the performance of end-to-end architectures for sound classification tasks. The receptive fields of the resulting feature maps are concatenated in a channel-wise manner. By using the Inception Nucleus, the network is given multiple convolutional layers with a variety of kernel sizes to capture richer features without additional tuning of hyperparameters.

Reshape. After applying 1D convolutions on the waveforms to obtain low-level features, the feature map, L, will be $\in \mathbb{R}^{1\times m\times n}$. We can treat L as a grayscale image with width=m, height=n, and channel=1. For simplicity, we transpose the tensor L to $L' \in \mathbb{R}^{m\times n\times 1}$. From here, we apply normal 2D convolutions with the VGG [73] standard

Table 7.2: Performance of CNNs on different subsets of the infant vocalization dataset. The first column denotes the experiment, the second column reports the results of the CNN with Inception, the third column reports the CNN without Inception Nucleus layer and the last column shows the performance at chance.

Model Comparison	CNN with Inception	CNN without Inception	Chance
Infant vs. Adult	94.12%	97.01%	50.00%
Vocalization vs. Non-Vocalization	90.49%	82.13%	50.00%
Canonical vs. Non-Canonical	76.17%	77.03%	50.00%
IDS vs. ADS	52.72%	60.28%	50.00%
Laugh/Cry vs. Can./Non-Can. vs. IDS/ADS	98.09%	97.86%	33.33%
Laugh/Cry vs. Can. vs. Non-Can. vs. IDS/ADS	74.69%	64.27%	25.00%
Laugh/Cry vs. Can. vs. Non-Can. vs. IDS vs. ADS	64.32%	41.34%	20.00%

Table 7.3: Classification accuracy is shown for the CNN with Inception Nucleus on three different comparisons as a function of age. # indicates the number of testing samples.

#	3-month	#	6-month	#	9-month	#	18-month	Model Comparison
228	71.05	219	84.93	15	100.00	72	95.83	Non-Vocalization vs. Vocalization
72	100.00	93	96.77	477	91.19	69	68.12	Canonical vs. Non-Canonical
123	53.16	210	62.86	15	60.00	63	63.37	Infant Directed vs. Adult Directed

kernel size of 3×3 and stride = 1 [73]. Also, the pooling layers have kernel sizes = 2×2 and stride = 2. By converting the acoustic waveform into a learned image representation, we are able to borrow deep learning techniques from the image processing literature, described next.

Conv2D. After the reshape layer, the receptive field can be treated as a grayscale image. We applied several 2D convolution layers similar to those used in the VGG network that is frequently used in the computer vision literature [73]. The kernel size and stride of the 2D convolutions was fixed to 3×3 and 1, respectively. Each 2D convolutional layer was followed by a max pooling layer with a kernel size of 2×2 and stride of 2.

Flatten and Fully Connected layers. The last convolution layer is followed with a co-called "flatten" layer which collapses the spatial dimensions to a single column vector that is passed to a fully connected layer. The number of units in the fully connected layer was set to the number of classes (which varied across models, see below) and the activation function of the final output layer was the *softmax* function.

7.4 Model Results

Models were trained on a total of 2456 audio clips, each being one second long. Recordings were down-sampled to 8 *kHz* and standardized to zero mean and unit variance. We shuffled the training data to enhance variability in the training set, and trained models using the Adam [41] optimizer. The optimizer is a variant of stochastic gradient descent that adaptively tunes the step size for each dimension. We used Glorot weight initialization [33] and trained each model with batch size 128 for up to 300 epochs until convergence. To avoid overfitting, all weight parameters were penalized by their ℓ_2 norm, using a λ coefficient of 0.0001. Our models were implemented in Keras [12] and trained using a GeForce RTX 2080 GPU.

Investigation of model parameters showed that bigger kernel sizes in the first layer lead to features that are more useful for classification, so larger sizes are reported in the following sections. Also, we found that deeper networks with larger numbers of parameters were less able to generalize learning to novel testing sets. Interestingly, this latter finding runs counter to results from the image recognition literature, in which deeper networks tend to generalize better than shallower ones [36, 39, 38]. The detriment of additional hidden layers may be attributable to the limited number of training examples, which can be tested in future studies with larger datasets. For the present study, we focused on smaller, more shallow networks with relatively fewer parameters.

7.4.1 Vocalizations vs. Infant Sounds

We combined the classes of canonical, non-canonical, and laugh/cry together to represent the "Infant" class, and combined IDS and other adult speech to represent the "Adult" class. We then trained our neural networks to distinguish between these two classes. The results are illustrated in the first row of Table 7.2. The model learned to discriminate these two perceptually distinct categories without difficulty, the use of an inception nucleus did not substantially influence performance.

7.4.2 Infant Vocalizations vs. Non-Vocalizations

Next we tested an ostensibly more challenging distinction between two broad classes of infant sounds. The "Vocalization" class contained canonical and non-canonical babbling whereas the "Non-Vocalization" class contained crying and laughing sounds. We trained both neural networks on the corresponding subset of training examples, and results are shown in the second row of Table 7.2. Performance dropped by 4-5% compared with the adult vs. infant model, which is consistent with the intuition that it more challenging to separate sounds made by infants. Also, the inception nucleus added eight percentage points to performance, indicating its potential utility in perceptually more challenging tasks.

7.4.3 Canonical vs. Non-Canonical Babbling

Next we tested an even finer-grained distinction between two basic kinds of infant vocalizations. Both classes contain vowels and consonants, with the main difference being their quality relative to adult speech and their sequential structure, or lack thereof. As shown in the third row of Table 7.2, model performance was still well above chance, but it dropped considerably compared with the prior two models, as expected. Interestingly, no benefit was conferred by the inception nucleus for distinguishing between different kinds of babbling. The two classes in this model were more homogeneous compared with classes in the previous two models, suggesting that the greater range of convolutional filters in the inception nucleus is more useful for representing relatively broad and diverse classes.

7.4.4 Infant Directed Speech vs. Adult Directed Speech

The models tested so far included classes that can be distinguished perceptually, based on gross differences in the properties of their sounds. Next we tested a distinction in vocalizations that is much more subtle in terms of differences in acoustic properties. In particular, we tested adult vocalizations that were categorized as either directed or not directed at the infant. So-called "motherese", a.k.a. infant-directed speech (IDS), tends to be more variable in intonation and intensity, and exaggerated in articulation, compared with adult-directed speech (ADS). However, these acoustic cues are highly variable within and between speakers, and it can be difficulty even for human listeners to perceive the differ-

ence. Model results are shown in the forth row of Table 7.2, and the expected difficulty of this distinction was born out in near-chance performance with the inception nucleus. Surprisingly, the model performed better without the inception nucleus, but recall that the inception nucleus was also not beneficial for distinguishing the two types of babbling. Two types of babbling and two types of adult speech are all relatively homogeneous categories, again suggesting that the greater range of convolutional filters in the inception nucleus is more useful for representing relatively broad and diverse classes.

7.4.5 Multiple Category Classification

The previous models that tested binary distinctions were useful for investigating the model in terms of its capability to learn different perceptual categories, and the effect of including the inception nucleus. However, for both theory and application, it is important to demonstrate an ability to distinguish multiple classes simultaneously. We ran three models that were trained to learn from three to five different classes simultaneously, as shown in the last three rows of Table 7.2. Results showed that performance was well above chance for all three models, and the inception nucleus was more beneficial as the number of classes increased, which corresponded with an increase in the diversity of sounds that needed to be classified.

7.4.6 Generalization on Samples Outside of our Training Set

Results presented thus far demonstrate the ability of CNN neural networks to learn a variety of classes of infant and adult vocalizations. In the models reported, training and testing sets were drawn from the same 15 families, which raises the question of whether learning would generalize to infants and adults not in the training set. We ran another set of models in which we tested performance by excluding recordings from one family in the training set, and testing performance on the untrained family. Performance dropped only slightly when generalizing to sounds from untrained infants and adults, by 3% at most. Therefore the model architecture and size and diversity of the data set enabled robust generalization.

7.4.7 Classification Performance as a Function of Age

Distinguishing different types of infant and adult vocalizations may be useful for infant monitoring applications, and analyses of learned representations may provide a data-driven method for studying how such complex sounds are processed and produced by human perceptual and motor systems. These analyses are left to future research, but our data set affords a relatively simple, initial test of the model as a tool for studying speech development. In particular, we can test performance as a function of infant age, and patterns of performance over time may provide information about changes in the sound structures of different vocalizations.

We examined classification performance for the CNN with Inception as a function of age for the three binary distinctions reported above. As shown in Table 7.3, results varied as a function of age, and the effect of age was different depending on the classes being distinguished. The strongest effect of age was for infant vocalizations versus non-vocalizations that became more distinct from each other as infants grew older. This result is consistent with the hypothesis that vocalizations become more clearly speech-like over the course of development. Interestingly, the opposite pattern was found for canonical versus non-canonical babbling. As speech develops, babbling becomes more and more canonical because it becomes more and more speech-like. Thus model results may have tracked the gradual extinction of non-canonical babble over the course of speech development. Finally, adult vocalizations were most difficult to discriminate when the infant was youngest at 3 months of age, suggesting that adults are less engaged in infant-directed speech when infants are less responsive at such an early age.

7.5 Summary

In this study, we developed, optimized, and tested CNNs with and without Inception Nucleus components, up to 17 layers deep, on end-to-end classification of infant sounds and vocalizations that were recorded in natural settings along with vocalizations and vocal interactions of caregivers and adults. Our results contribute to the machine learning community as well as the developmental speech science community by showing how deep learning techniques developed for speech and image recognition can be leveraged for speech devel-

opment. Further research is needed to control for the number of training and testing examples per class, and confirm that results as a function of age were not unduly influenced by differences in the numbers of training and testing examples. Further work is also needed to test whether the model may shed light on differences between normal and disordered speech development, and whether important aspects of these differences might be revealed through comparisons of learned representations in models trained on speech from different populations. We believe that models like those presented in this chapter are opening new avenues toward diagnostic tools for measuring normal vs. abnormal speech development, as well as research tools for examining the developmental time course of speech.

Chapter 8

Conclusion and Future Work

8.1 Summary

This dissertation investigates computer perception in both vision and audition, specifically, in object detection and auditory object recognition. To address these challenging problems, we received inspiration from the visual pathways of the human brain to propose a novel object detection framework, as well as an end-to-end auditory object recognition system. Chapter 2 introduced an object detection method by interpreting the connection weights of the network. This approach uses the gradients of the labels with respect to the input pixels to find the location of the main object in the image. Toward this end, we formulated the problem as a sensitivity analysis, and evaluated it on state-of-the-art object detection benchmarks. Our results demonstrated the powerful performance of our proposed algorithm in comparison of the state-of-the-art methods.

In Chapter 3, we took a further step by proposing a fundamentally novel framework for object detection. We noticed that there are two main issues in current object detection frameworks. The first involves the lack of having a selective attention mechanism. The second is not assigning dual neural networks for solving the detection tasks. The proposed algorithm benefits from selective attention as well as dual neural networks in object detection. By utilizing ventral and dorsal neural networks, we obtained more accurate and tight bounding boxes around all objects of interest.

In Chapter 4, we further explored the importance of dual neural networks in object detection by proposing a more sophisticated selective attention approach. The proposed

method benefits from all intermediate layers of a "Ventral" ("What") neural network by extracting weighted attention map information from all the hidden layers and densely stacking them together. This approach results in more accurate attentional maps due to the use of location information from earlier layers as well as obtaining semantically meaningful information from deep layers. The evaluation of this model exhibits strong performance when compared to state-of-the-art methods.

In Chapter 5, we explored human attention vs. CNN attention. We recruited human subjects from the population of UC Merced undergraduate students and we used an eye-tracker while they looked at pre-selected images and named all objects aload. After comparing human attention to deep neural network attention, we noticed that human attention is sharp and far from deep neural attention.

We also found our approach sometimes failed to direct adequate attention to small objects.

Starting with Chapter 6, we proposed a novel architecture for auditory object recognition without the need for feature extraction. The proposed architecture takes raw audio streams and maps them to labels without the use of hand engineered features. We proposed a novel inception nucleus which contains a number of 1D convolutions for processing these raw signals. We evaluated this method in comparison to the state-of-the-art approaches on available benchmarks. Our proposed method showed a significant improvement over the other approaches. We also analyzed the low level and high level filters to obtain a better understanding of our proposed architecture. We observed that our model learns wavelet-like filters in earlier layers and that high level representations are perceptually meaningful.

In Chapter 7, we further analyzed our end-to-end auditory object recognition network on a new dataset of infant voices, including vocalization and non-vocalization babblings, adult directed and infant directed speeches, and cry/laughing sounds. We learned that our network performs significantly well even on hard tasks like canonical sounds vs. non-canonical babbling. We believe that models like those presented in this chapter are opening new avenues toward diagnostic tools for measuring normal vs. abnormal speech development, as well as research tools for examining the developmental time course of speech.

8.2 Future Work

In this part, we present two interesting topics for future development.

8.2.1 Selective Attention in Auditory Domain

As stated in the previous chapters, selective attention can boost the performance of object detection frameworks by reducing the search space for objects of interest. One potential extension of selective attention work can be done in auditory domain by implementing selective attention mechanisms to reduce the distractions in audio space with the aim of boosting the performance in audio recognition systems.

8.2.2 Cross-Modal Object Recognition

To further improve the efficiency and accuracy of object recognition networks, one potential solution involves analyzing both the visual field and the auditory filed. For instance, if one wants to recognize objects in a dark video/image, in which the objects are not clearly visible, listening to the sound of events might boost the performance of the detection/recognition algorithm. Also, cross-modal learning might help to learn richer features due to the collaboration of two networks together, and this might lead to a more accurate recognition system.

8.2.3 Retrieving an Object Given the Sound of the Object

By advancement of deep neural networks, it is predictable that we should be able to retrieve objects given the sound of them. For instance, given a sound of dog barking, are we able to retrieve an image of a dog's body? One can consider this technique as computer imagination. Listening to an audio can result in imagining the visual concept of the audio.

8.3 Conclusion

The work presented in this dissertation offers new approaches to computer perception in both visual and auditory modalities. Contributions have focused on implementations of selective attention and on the ability of end-to-end learning. While our explorations

have found that computer perception and human perception differ in important ways, there are still opportunities to improve our best algorithms by taking inspiration from biological systems that support awareness of the world.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu. Convolutional neural networks for speech recognition. *ASLP*, 2014.

[3] H. F. Alaie, L. Abou-Abbas, and C. Tadj. Cry-based infant pathology classification using gmms. *Speech communication*, 2016.

[4] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. In *NIPS*, 2016.

[5] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016.

[6] P. Blouw, X. Choo, E. Hunsberger, and C. Eliasmith. Benchmarking keyword spotting efficiency on neuromorphic hardware. In *NCEW*, 2019.

[7] D. H. Brainard and S. Vision. The psychophysics toolbox. *Spatial Vision*, 10:433–436, 1997.

[8] E. H. Buder, A. S. Warlaumont, and D. K. Oller. An acoustic phonetic catalog of prespeech vocalizations from a developmental perspective. *Comprehensive perspectives on child speech development and disorders: Pathways from linguistic theory to clinical practice*, 4:103–134, 2013.

[9] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

[10] S. Chachada and C.-C. J. Kuo. Environmental sound recognition: A survey. *APSIPA*, 2014.

[11] M. Cho, S. Kwak, C. Schmid, and J. Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *CVPR*, 2015.

[12] F. Chollet et al. Keras. `https://keras.io`, 2015.

[13] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017.

[14] K. J. Dai and Y. L. R-FCN. Object detection via region-based fully convolutional networks. arxiv preprint. *arXiv preprint arXiv:1605.06409*, 2016.

[15] W. Dai, C. Dai, S. Qu, J. Li, and S. Das. Very deep convolutional neural networks for raw waveforms. In *ICASSP*, 2017.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[17] R. Desimone and J. Duncan. Neural mechanisms of selective visual attention. *Annual review of neuroscience*, 18(1):193–222, 1995.

[18] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

[19] M. Ebrahimpour, S. Schneider, D. Noelle, and C. Kello. Infantnet: A deep neural network for analyzing infant vocalizations. In *INTERSPEECH*, 2020.

[20] M. Ebrahimpour, T. Shea, A. Danielescu, D. Noelle, and C. Kello. End-to-end auditory object recognition via inception nucleus. In *ICASSP*, 2020.

[21] M. K. Ebrahimpour, B. Falandays, S. Spevack, and D. C. Noelle. Do humans look where deep convolutional neural networks "attend"? In *2019 International Symposium on Visual Computing (ISVC)*.

[22] M. K. Ebrahimpour, J. B. Falandays, S. Spevack, M.-H. Yang, and D. C. Noelle. Ww-nets: Dual neural networks for object detection. *arXiv preprint arXiv:2005.07787*, 2020.

[23] M. K. Ebrahimpour, J. Li, Y.-Y. Yu, J. Reesee, A. Moghtaderi, M.-H. Yang, and D. C. Noelle. Ventral-dorsal neural networks: Object detection via selective attention. In *WACV*.

[24] M. K. Ebrahimpour and D. C. Noelle. On the interpretability of deep neural networks: Object localization with sensitivity analysis. In *2019 International Symposium on Visual Computing (ISVC)*.

[25] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[28] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[29] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware CNN model. In *CVPR*, 2015.

[30] J. Gilkerson and J. A. Richards. The lena natural language study. *Boulder, CO: LENA Foundation. Retrieved March*, 2008.

[31] R. Girshick. Fast R-CNN. In *CVPR*, 2015.

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

[33] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010.

[34] R. Gokberk Cinbis, J. Verbeek, and C. Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014.

[35] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *CVPR*, 2017.

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

[37] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[38] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*, 7, 2017.

[39] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[40] J. J. Huang and J. J. A. Leanos. Aclnet: efficient end-to-end audio classification cnn. *arXiv preprint arXiv:1811.06669*, 2018.

[41] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[42] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *CVPR*, 2016.

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

[44] D. Li, J.-B. Huang, Y. Li, S. Wang, and M.-H. Yang. Weakly supervised object localization with progressive domain adaptation. In *CVPR*, 2016.

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

[46] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

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

[48] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[49] A. Nguyen, J. Yosinski, and J. Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616*, 2016.

[50] R. C. O'Reilly and Y. Munakata. *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain*. MIT press, 2000.

[51] B. Peter and A. A. MacLeod. *Comprehensive perspectives on speech sound development and disorders: Pathways from linguistic theory to clinical practice*. Nova Science Publishers, Inc., 2013.

[52] K. J. Piczak. Environmental sound classification with convolutional neural networks. In *MLSP*, 2015.

[53] J. Pons and X. Serra. Randomly weighted cnns for (music) audio classification. In *ICASSP*, 2019.

[54] G. M. Pretzer, L. D. Lopez, E. A. Walle, and A. S. Warlaumont. Infant-adult vocal interaction dynamics depend on infant vocal type, child-directedness of adult speech, and timeframe. *Infant Behavior and Development*, 2019.

[55] K. Rajaei, Y. Mohsenzadeh, R. Ebrahimpour, and S.-M. Khaligh-Razavi. Beyond core object recognition: Recurrent processes account for object recognition under occlusion. *bioRxiv*, page 302034, 2018.

[56] M. Ravanelli and Y. Bengio. Speaker recognition from raw waveform with sincnet. In *SLT*, 2018.

[57] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *CVPR*, 2014.

[58] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[59] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.

[60] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

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

[62] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun. Object detection networks on convolutional feature maps. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 39(7):1476–1481, 2017.

[63] Y. D. Rosita and H. Junaedi. Infant's cry sound classification using mel-frequency cepstrum coefficients feature extraction and backpropagation neural network. In *ICST*. IEEE, 2016.

[64] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[65] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*. MIT Press, 1986.

[66] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *ICASSP*, 2015.

[67] J. Salamon, C. Jacoby, and J. P. Bello. A dataset and taxonomy for urban sound research. In *ICM*, 2014.

[68] J. Sang, S. Park, and J. Lee. Convolutional recurrent neural networks for urban sound classification using raw waveforms. In *EUSIPCO*, 2018.

[69] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

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

[71] Z. Shi, T. M. Hospedales, and T. Xiang. Bayesian joint topic modelling for weakly supervised object localisation. In *CVPR*, 2013.

[72] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[73] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[74] P. Siva, C. Russell, and T. Xiang. In defence of negative mining for annotating weakly labelled data. In *ECCV*. Springer, 2012.

[75] I. Sobol. Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments*, 1(4):407–414, 1993.

[76] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[77] D. Stowell and M. D. Plumbley. Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning. *PeerJ*, 2014.

[78] K. Tang, A. Joulin, L.-J. Li, and L. Fei-Fei. Co-localization in real-world images. In *CVPR*, 2014.

[79] Y. Tokozume and T. Harada. Learning environmental sounds with end-to-end convolutional neural network. In *ICASSP*, 2017.

[80] Z. Tüske, P. Golik, R. Schlüter, and H. Ney. Acoustic modeling with deep neural networks using raw time signal for lvcsr. In *ACISCA*, 2014.

[81] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[82] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *NIPS*, 2013.

[83] H. Xu, X. Lv, X. Wang, Z. Ren, N. Bodla, and R. Chellappa. Deep regionlets for object detection. In *ECCV*, 2018.

[84] D. L. K. Yamins, H. Hong, C. F. Cadieu, E. A. Solomon, D. Seibert, and J. J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

[85] J. Yang and M.-H. Yang. Learning hierarchical image representation with sparsity, saliency and locality. In *BMVC*, 2011.

[86] J. Yang and M.-H. Yang. Top-down visual saliency via joint crf and dictionary learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(3):576–588, 2017.

[87] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[88] A. Zabidi, L. Y. Khuan, W. Mansor, I. M. Yassin, and R. Sahak. Classification of infant cries with asphyxia using multilayer perceptron neural network. In *ICCEA*, 2010.

[89] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014.

[90] X. Zhang, T. Wang, J. Qi, H. Lu, and G. Wang. Progressive attention guided recurrent network for salient object detection. In *CVPR*, 2018.

[91] Y. Zhang, Y. Bai, M. Ding, Y. Li, and B. Ghanem. W2f: A weakly-supervised to fully-supervised framework for object detection. In *CVPR*, 2018.

[92] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016.

[93] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu. Scale-transferrable object detection. In *CVPR*, 2018.