Towards Human-Centered AI-Powered Assistants for the Visually Impaired

by

Linda Wang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2020

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The following four papers are used in this thesis. I was co-author with major contributions to the design, analysis, writing and editing.

L. Wang and A. Wong, "Implications of Computer Vision Driven Assistive Technologies Towards Individuals with Visual Impairment," *Workshop on Fairness Accountability Transparency and Ethics in Computer Vision (FATE/CV)*, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
This paper is incorporated in Chapter 1.

L. Wang, A. Patnaik, E. Wong, J. Wong and A. Wong, "OLIV: An Artificial Intelligence-Powered Assistant for Object Localization for Impaired Vision," *Computational Vision and Imaging Systems (CVIS)*, vol. 4, no. 1, 2018.
This paper is incorporated in Chapter 3.

L. Wang and A. Wong, "Enabling Computer Vision Driven Assistive Devices for the Visually Impaired via Micro-architecture Design Exploration," *Women in Computer Vision Workshop (WiCV)*, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
This paper is incorporated in Chapter 4.

L. Wang, M. Famouri and A. Wong, "DepthNet Nano: A Highly Compact Self-Normalizing Neural Network for Monocular Depth Estimation," *Women in Computer Vision Workshop (WiCV)*, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
This paper is incorporated in Chapter 5.

Abstract

Artificial intelligence has become ubiquitous in today's society, aiding us in many everyday tasks. Given particular prowess of today's AI technologies in visual perception and speech recognition, an area where AI can have tremendous societal impact is in assistive technologies for the visually impaired. Although assisting the visually impaired for tasks such as environment navigation and item localization improves independence and autonomy, concerns over privacy arise. Taking privacy of personal data into consideration, we present the design of a human-centered AI-powered assistant for object localization for impaired vision (OLIV). OLIV integrates multi-modal perception (custom-designed visual scene understanding and speech recognition and synthesis) for the purpose of assisting the visually impaired in locating misplaced items in indoor environments.

OLIV is comprised of three main components: speech recognition, custom-designed visual scene understanding, and synthesis. Speech recognition allows these individuals to independently query and interact with the system, increasing their level of independence. Visual scene understanding performs on-device object detection and depth estimation to build up a representation of the surrounding 3D scene. Synthesis then combines the detected objects along with their locations and depths with the user's intent to construct a verbal semantic description that is verbally conveyed via speech synthesis.

An important component of OLIV is scene understanding. Current state-of-the-art deep neural networks for the two tasks have been shown to achieve superior performance, but requires high computation and memory, making them cost prohibitive for on-device operation. On-device operation is necessary to address privacy concerns related to misuse of personal data. By performing on-device scene understanding, data captured by the camera will remain on the device. To address the challenge of high computation and memory requirements, two different architecture design exploration approaches, micro-architecture exploration and human-machine collaborative design strategy, are taken to design efficient neural networks with an optimal trade-off between accuracy, speed and size. Micro-architecture exploration approach resulted in a highly compact single shot network architecture for object detection. Human-machine collaborative design strategy resulted in a highly compact densely-connected encoder-decoder network architecture for monocular depth estimation. Through experiments on two indoor datasets to simulate environments OLIV operates in, the object detection network and depth estimation network were able to achieve CPU speeds of 17 FPS and 9.35 FPS, sizes of 6.99 and 3.46 million parameters, respectively, while maintaining comparable accuracy performance. Size and speed are important for on-device scene understanding on OLIV to provide a more private assistance for the visually impaired.

Acknowledgements

Firstly, I would like to thank my supervisor, Prof. Alexander Wong. Thank you for being an amazing mentor and helping me grow as a researcher. Your passion for using your expertise in computer vision to help others has truly been inspirational and I hope to carry this forward as I begin my career in computer vision.

Secondly, I would like to thank Prof. Boger and Prof. Scott for reviewing my thesis. Taking the time out of your busy schedules, especially during the current pandemic, to read and revise my thesis is greatly appreciated.

I would also like to thank my family and friends for their constant support and curiosity about my thesis and other research projects. In addition, I would like to thank my fourth year design group (W3P), as that is where the idea for this thesis first originated. I would also like to thank everyone in the Vision and Image Processing Lab. I've enjoyed collaborating on different projects apart from my thesis, which has broaden the scope of my research experience.

Lastly, I would like to thank Microsoft for providing Azure resources for my research in this thesis.

Dedication

I would like to dedicate this thesis to my parents for being my biggest cheerleaders all the way to the end of my educational journey (...unless I pursue a PhD in the future). I would also like to dedicate this thesis to my grandparents, whose generation can benefit greatly from human-centered artificial intelligence solutions.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, the rise of artificial intelligence (AI) has made previously unsolvable tasks possible, particularly in the areas of visual perception, speech recognition and synthesis. As AI grows in ubiquity in society, larger issues regarding the use of AI need to be considered. An important area to consider is AI-powered assistive technologies for the visually impaired. With the use of deep learning, visual perception tasks, such as image classification, object detection and monocular depth estimation, are able to be achieved and have the potential to aid these individuals in tasks requiring the identification of objects and navigation.

Vision impairment and blindness affect 2.2 billion of the world's population, of which 1 billion includes those with moderate or severe distance vision impairment or blindness [37]. Severe vision impairment or blindness for those individuals cause a significant amount of burden when conducting their daily tasks [28]. AI-powered assistive technologies show the potential to reduce some burden placed on these individuals. By assisting visually impaired individuals with tasks they would otherwise need help in or struggle with, their level of independence and autonomy increase.

Although there are positive aspects leveraging AI to assist the visually impaired, there are also negative aspects that should be addressed. The use of black box AI solutions raises many concerns such as fairness and bias of the models, which often stem from training data sample sets that are not representative of the general population and exclude certain groups or characteristics [34]. There are also ethical concerns related to privacy protection as many visual perception algorithms rely on camera input. As such, it is crucial to address issues related to the implications of AI in the development of AI-driven assistive technologies. In this thesis, the aim is to explore the privacy concerns related to the deployment of computer vision based assistive technologies.

1.1 Current Computer Vision Based Solutions

One area assistive technologies have become an integral part in the lives of those with visual impairment is overcoming barriers faced in everyday life. These individuals face adversity in all stages of life. For instance, severely visually impaired young people use their information and communication technology (ICT) device as more than just a device to overcome environmental barriers but also a means of communication for peers in their school [47]. However, the use of ICT devices is found to symbolize restriction and difference. In order fit in as ordinary young people, those who are capable to participate without their device choose not to use it [47].

The ever growing presence of smartphones and advancements in computer vision are transforming the accessibility of assistive technologies, allowing individuals to overcome social barriers and have autonomy over when and how they access information. Smartphone applications, such as SeeingAI and Lookout, use auditory cues to assist users in identifying scenes, recognizing faces, reading short text, documents and currency. These assistive technologies also allow these individuals to have a sense of safety, security and privacy by notifying how many people are in the surroundings [23, 9].

Individuals with visual impairment also face difficulty localizing themselves in unknown indoor and outdoor environments. Research projects are using cameras and sensors to give directions so these individuals can navigate outdoor and indoor environments independently. For instance, a prototype was developed for guiding the visually impaired across streets in a straight line using a wearable computer-based orientation and wayfinding aid [43]. The results showed that all the prototypes showed significant improvement in their veering performance, with average veer reduced by 31 percent from the baseline veer [43]. Based on the results, this prototype enabled these individuals to independently and confidently cross the street, knowing that they will be signaled if veering off. For indoor navigation, Tian et al. developed a proof of concept computer-vision based indoor wayfinding aid that detects doors and elevators, as well as text on signs, to find different rooms [50]. Using geometric door models and corner detection to detect doors, as well as text extraction, localization and recognition for door signs, the model was able to achieve high true-positive rates of 89.5% and 92.3% for both protruding and non-protruding object detection respectively. The model was also able to recognize door signage with 71% accuracy.

1.2 Privacy Concerns

Computer vision based assistive technologies for the visually impaired allow these individuals to gain independence and autonomy over different aspects of their life. However, these devices also pose privacy risks because of the vast amounts of personal data stored. Although individuals with visual impairment felt that smartphones help them communicate and achieve greater independence, these devices create privacy risks because of the amount of personal data stored and shared in the cloud for processing. As well, their poor visual acuity makes it hard to safeguard their information, such as if someone is around and eavesdropping [1].

Home-monitoring for older adults, who represent majority of those with visual impairment [37], reliefs caregivers burden and allows individuals with severe visual impairment to live independently, but the devices for monitoring also store personal data. Studies have found that older adults are willing to have activity monitoring shared with family members and doctors if the collected data is useful, but expressed that the greatest concern is exploitation and misuse of their personal health information [25].

Based on the studies, the greatest fear associated with the collection of personal data is the concern that their collected data could end up in the wrong hands and be misused. In addition to the fear of personal information being exploited, the use of cameras is obtrusive and found to elicit greater fears than wearable solutions. In a comparison of four ambient intelligent systems, the camera-based behaviour and emergency detection system was perceived with the greatest fear and highest level of concern [25]. However, studies have also shown that there is a tradeoff between gained autonomy and privacy costs. Older adults with lower levels of functioning are willing to accept video cameras and tradeoff the privacy lost if camera-based solution could prevent transfer to a long term care facility [51].

The different perceptions of privacy over the use of data, as well as the potential benefits of using cameras for home monitoring, suggest that privacy is a complex topic. In this thesis, the concern of personal data exploitation and privacy from camera-based solutions are addressed.

1.3 Proposed Framework and Thesis Contributions

Artificial intelligence has the potential to impact people's lives. However, accuracies of computer vision models are insufficient for assistive technologies, which interact with and around humans. Recently, the term *human-centered artificial intelligence* is used to refer

to intelligent systems that are aware of the interaction with humans and are designed with social responsibility in mind [42]. The purpose of this thesis is to propose the design of OLIV (object localization for impaired vision), an AI-powered assistant for assisting the visually impaired in locating misplaced items in indoor environments, which has not previously been explored. To design OLIV, a human-centered AI design strategy was taken to ensure the potential privacy implications posed are considered in the design. There are three main contributions:

1. A novel human-centered end-to-end artificial intelligence-powered assistant system designed to aid individuals with impaired vision in their day-to-day tasks in locating displaced objects.
2. A highly compact single-shot network architecture for on-device object detection.
3. A highly compact densely-connected encoder-decoder network architecture for on-device monocular depth estimation.

The goal of the three main contributions in this paper aims to assist the visually impaired in locating displaced items while addressing privacy concerns related to the misuse of personal data and privacy lost from cameras. By performing on-device scene understanding, data captured by the camera will remain on the device. In order to perform on-device scene understanding, the neural network architectures must be compact to fit on an edge or mobile device, which are more affordable than graphical processing units (GPUs) and are widely adopted. In addition to size, the neural networks must also be fast for a seamless user interaction.

In this thesis, a review of relevant background concepts are presented in Chapter 2. A system overview of OLIV is detailed in Chapter 3. Problem formulation, methods, and results of the designed efficient neural networks for object detection and depth estimation are detailed in Section 4 and 5, respectively. Lastly, conclusions, recommendations, and future works are discussed in Chapter 6.

Chapter 2

Background

In this chapter, background information for each proposed contribution are presented. The background on object detection and the chosen object detection model for the OLIV system is detailed in Section 2.1. An overview of monocular depth estimation network architecture components are described in Section 2.2. A brief description of existing methods to create efficient neural networks are discussed in Section 2.3. The method used to create the efficient depth estimation network for OLIV is further explained in Section 2.3.2.

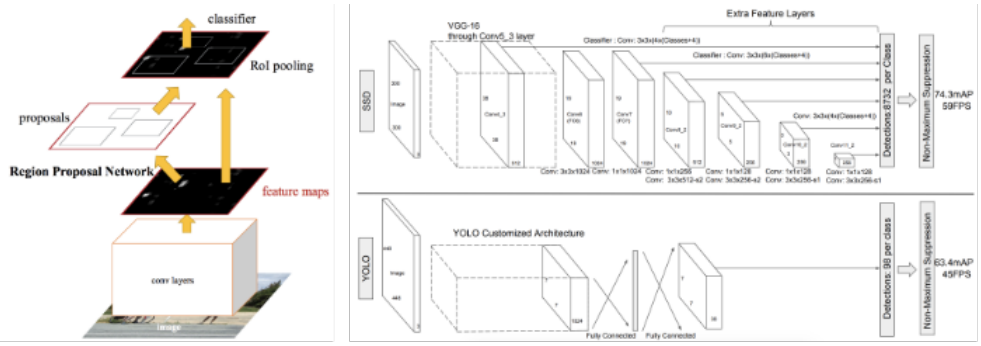
2.1 Efficient Object Detection

Deep learning has shown significant progress in the performance of computer vision tasks, such as object detection, text recognition and image classification. However, many of the technologies that use computer vision neural networks are internet-connected devices so images can be processed in the cloud. This makes the technology limited and with the increasing presence of mobile devices, users should be able to use the technology anywhere, regardless of internet connectivity. Neural networks for object detection, such as Faster-RCNN Inception ResNet [20], achieve high performance, however deploying for real-time on device and edge applications are difficult due to limited computation, power and space.

Object detection contains two parts: i) the feature extractor, which extracts salient features from the image, and ii) the meta-architecture, which makes detection predictions based on the features. In order for the object detection network to be computationally efficient, both the meta-architecture and feature extractor must be efficient. In this thesis, MobileNetV2 and single-shot detector (SSD) were used as the feature extractor and meta-architecture, respectively. These are summarized in the following subsections.

2.1.1 Single Shot Detector

To date, single shot detectors are the fastest compared to other meta-architectures that use region proposal networks, such as Faster R-CNN and R-FCN [41, 11]. This is because region proposal networks (RPN) require two shots to detect objects in an image. The first pass generates regional proposals. The second shot then detects objects for each proposal, as shown on the left in Fig. 2.1. To generate proposals for regions of where there may be objects, a sliding window with k anchors is slid over a feature map. For each proposal, the classifier determines the probability of a proposal having a target object. The regressor then regresses the coordinate of the proposal to better fit the object [41].



a) Region Proposal Network b) Single Shot Meta Architectures

Figure 2.1: Region proposal network, taken directly from Ren et al. [41], vs. single shot meta architectures, taken directly from Liu et al. [33]. (Left) Region proposal network architecture used in object detection networks such as Faster R-CNN [41], (right) single shot meta architectures found in SSD (top) and YOLO (bottom) [33].

Single shot meta architectures eliminates the use of RPN and detects the image in one shot. The right image of Fig. 2.1 shows the network architectures for two different single shot meta architecture designs. The bottom single shot meta architecture design is YOLO. This network divides the images into an $S \times S$ grid, where S is the number of cells, so that a single neural network can predict bounding boxes and class probabilities in one evaluation [40]. However, due to the image being split into grids, YOLO tends to miss smaller objects [33], resulting in lower accuracy than the previous models.

SSD also detects objects in one shot. To make up for the loss in accuracy, SSD introduces multi-scale features and default boxes [33]. As shown in the top image in Fig. 2.1, detections are based on multiple feature map layers, whereas for YOLO, detections are only based on one layer. For certain layers, each pixel of the feature map contains

k bounding boxes. Each bounding box contains a different size and aspect ratio since objects are different shapes and sizes. For each bounding box, class scores and offsets are computed. These improvements allow SSD to run in real-time on GPU with competitive accuracy to Faster R-CNN [33].

On three different object detection datasets, PASCAL VOC, ILSVRC DET and MSCOCO, SSD achieved state of the art accuracy results over Faster R-CNN and YOLO. By eliminating the use of RPN, SSD can process 6 times more frames per second (FPS) than Faster R-CNN [33]. SSD has higher accuracy, as well as higher FPS, which shows promise as an efficient object detection meta-architecture.

2.1.2 MobileNet

In order to make neural networks more efficient at extracting features, the MobileNet family introduced depthwise separable convolution. Depthwise separable convolution is comprised of a depthwise convolution followed by a pointwise convolution, as shown on the left in Fig. 2.2. For depthwise convolution, convolution is performed for each channel separately, then pointwise convolution is used to learn about the channels in order to change the dimension [44]. For depthwise separable convolution, the computational cost for the number of operations is:

$$h_i \cdot w_i \cdot d_i(k^2 + d_j) \tag{2.1}$$

where k, h_i, w_i are the kernel and feature map height and width sizes, respectively. d_i and d_j are the number of input and output channels. For standard convolution, the computation cost is:

$$h_i \cdot w_i \cdot d_i \cdot d_j \cdot k^2 \tag{2.2}$$

By using depthwise separable convolutions, the reduction in computation cost is $\frac{1}{d_j} + \frac{1}{k^2}$. This means that if the kernel size is 3×3, 9 times less computation is needed with only a small drop in accuracy.

In addition to depthwise separable convolution, MobileNet also includes two additional hyper-parameters, width and resolution multiplier to further improve the efficiency of the network. The width multiplier, α, reduces the width of the network uniformly at each layer by multiplying the number of input channels and output channels by α. The resolution multiplier, ρ, reduces the representation by multiplying each feature map by ρ [44]. With the width multiplier and resolution multiplier, the computational cost now becomes:

$$\rho h_i \cdot \rho w_i \cdot \alpha d_i(k^2 + \alpha d_j) \tag{2.3}$$

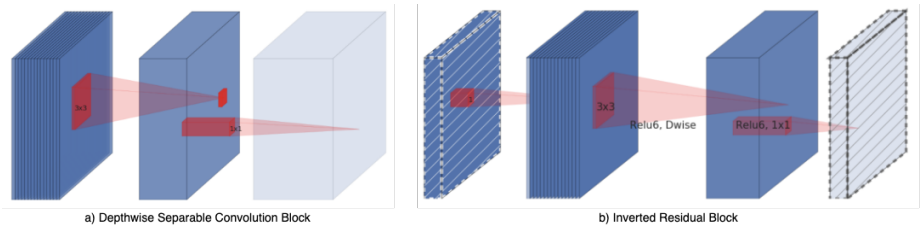


Figure 2.2: MobileNet building blocks taken directly from Sandler et al. [44]. (Left) depthwise separable convolution block introduced in MobileNetV1, and (right) inverted residual structure introduced in MobileNetV2 [44].

To account for the drop in performance using depthwise separable convolutions, MobileNetV2 introduced the inverted residual structure, shown on the right in Fig. 2.2. Before the depthwise separable convolution layer, pointwise convolution is used to expand the low-dimensional feature map to a higher dimension by a factor t. Depthwise convolution is then performed on the higher dimensional feature maps and then pointwise convolution is used to project the feature maps back to the lower dimension [44]. The final computational cost is now:

$$\rho h_i \cdot \rho w_i \cdot \alpha d_i \cdot t(\alpha d_i + k^2 + \alpha d_j) \tag{2.4}$$

Compared to other compact efficient architectures, such as MobileNetV1, ShuffleNet and NasNet-A, MobileNetv2 obtained the best results on ImageNet classification while also achieving the fastest CPU running time [44]. ImageNet is a large scale image dataset representing over 5247 categories [12] and is widely used as a benchmark to compare performances of different architectures. The performance of MobileNetV2 was also compared as a feature extractor for mobile and real-time meta-architectures of SSD, SSDLite and YOLOv2, where MobileNetV2 achieved the best results with SSD [44]. In addition, MobileNetV1 with SSD has shown promising results in terms of CPU running time compared to other network architectures in [20]. Based on the results of MobileNetV2, this network is chosen as the feature extractor for the SSD meta-architecture.

2.2 Monocular Depth Estimation

The task of estimating depth from 2D images is crucial for applications such as 3D scene understanding and reconstruction. In recent years, monocular depth estimation, where a dense depth map is obtained from a single image has gained traction. Compared to depth

estimation from stereo images or video sequence, monocular depth estimation is an ill-posed problem, which means there are more than one possible unique solution. To tackle this ill-posed problem, one method that has shown promise is deep convolutional neural networks (DCNNs).

2.2.1 Encoder-Decoder Architecture

These DCNNs learn deep features to estimate a depth for each pixel. Most of the recent developments have focused on an encoder-decoder architecture, such as the architecture in Fig. 2.3, with very powerful deep neural network encoder macroarchitecture designs, such as VGG, ResNet and DenseNet [15, 2, 29], to learn deep features.

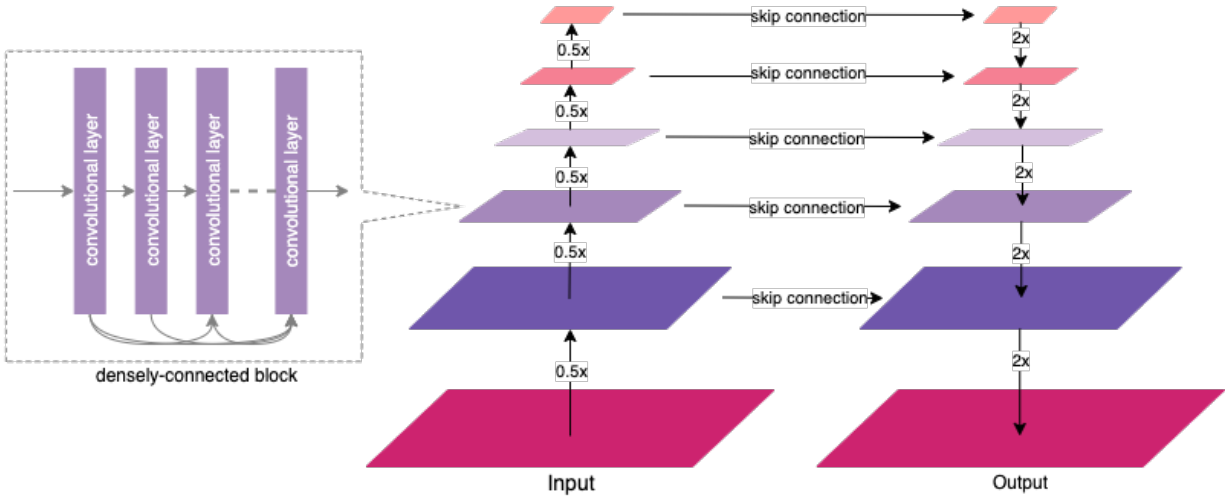


Figure 2.3: Encoder-decoder architecture with skip connections for monocular depth estimation. The encoder contains densely-connected blocks to alleviate training.

The encoder layers in such an architecture are designed to learn a multitude of low to mid level features for characterizing an input scene. Next, the decoder layers are designed to merge and upsample the features learned from the encoder layers to recover a dense depth map. The decoder layers consist of upsampling blocks followed by a concatenation and two convolutional operations. Finally, the encoder-decoder architecture follows with a convolutional layer at the end that is designed to produce the final dense depth map.

2.2.2 Skip Connections

In general, as the number of layers increase, the network accuracy improves because each layer is learning deeper features. However, He et al. [18] found that in general, after a certain depth, network accuracy starts to decrease. For instance, He et al. [18] demonstrated that a 56-layer deep convolutional neural network has higher training and test error than a 26-layer CNN deep convolutional neural network. To alleviate training of deep neural networks, He et al. introduced a fundamental buiding block, known as the residual block. The residual block adds a previous layer to the current layer. By adding information from previous layers, the network can learn residuals or errors between a previous layer and the current one. Extending upon this idea of skip connections, densely-connected network architectures consists of a large number of skip connections between different layers, as shown in Fig. 2.3. More specifically, instead of adding the previous layer as an identity function, densely-connected architectures concatenate outputs from previous layers to the current layer. This is found to alleviate the vanishing gradient problem, strengthen feature propagation and feature reuse [19]. As such, the introduction of encoder-encoder skip connections into a deep encoder-decoder architecture can improve the training process and improve network performance.

Furthermore, in the case of encoder-decoder architectures, as the layers in the encoder get deeper, higher level features are learned, however, the resolution of the feature maps get progressively lower [6]. As such, the input to the decoder is of low resolution. Since the purpose of the decoder network is to upsample the features learned from the encoder, the resulting depth map image would also have low resolution.

To overcome the low resolution problem, an effective strategy is the leveraging of skip connections between encoder and decoder layers within an encoder-decoder architecture. Such encoder-decoder skip connections merge high resolution feature maps from the encoder layers to the features in the decoder layers, resulting in a more detailed decoder output [7].

2.3 Methods to Create Efficient Neural Networks

In recent years, neural network architecture designs have been getting deeper to fit the large amounts of data, such as ImageNet. Although these bigger networks have better performance than networks with fewer layers, not every parameter in the network is utilized. For many networks, low resource utilization is caused by stochastic gradient descent (SGD) and initialization [38, 14, 17]. In addition, the network architecture may also not be suitable

for the given task. For instance the number of channels and depth of network may not be properly configured for the domain. As a result, there are inefficiencies in computational resources and the full potential of the model is not realized. In this section, two methods to create efficient neural networks are addressed.

2.3.1 Network Pruning

Neural network pruning addresses the under utilization of network resources and introduce methods to optimize the efficiency of neural networks without decreasing performance.

One approach by Han et al. [17], introduced a training approach to improve the efficiency of the network by removing connections with small weights and neurons with zero input or output connections. Frankle et al. [14] introduced the lottery ticket hypothesis, where p percent of the weights with the lowest magnitude are set to 0 while the rest of the network is set to initial random initialization to be retrained. Instead of removing neurons, Qiao et al. [38], introduced an addition to the standard SGD optimization method that re-allocates these dead neurons to different parts of the network and re-initializes the weights for continued training in order to realize the full potential of the network.

The methods introduced follow similar procedures. First, a threshold is determined to distinguish which connections and neurons to keep or remove. Once the weights are below the determined threshold, the neural network architecture is modified to be more resource efficient. After the architecture has been modified, the training scheme is also adjusted so that the network can continue to learn with the remaining or new connections.

Based on experiments on MNIST and ImageNet, the results showed that [17] can decrease the size and computation of various neural networks without decreasing accuracy. For instance, pruning LeNet-300-100 and LeNet-5 on MNIST, reduced the number of weights by 12x and computation by 12x and 6x, respectively. Without impacting accuracy, the number of weights and computations for AlexNet and VGG-16 on ImageNet were able to be reduced by 9x and 3x, 12x and 5x, respectively. Pruning on VGG-16 also showed that the largest fully-connected layers can be pruned to less than 4% of their original size. This indicates that not all the connections in the fully-connected layers were necessary.

Similarly, [38] also trained various sizes of networks on ImageNet and CIFAR. Using the CIFAR-10/100 datasets, neural rejuvenation can achieve the same accuracy as baseline while maintaining half the parameters for VGG-19, ResNet-164 and DenseNet-100-40. In addition to achieving the same accuracy as the full network, the results also showed that by re-allocating neurons to different parts of the network, neural rejuvenation can in fact increase accuracy. For instance, on ImageNet, VGG-16 with neural rejuvenation achieve

23.71% top-1 error with just 21.5M parameters, which is better than the baseline of 36.66% top-1 error with 23.2M parameters.

2.3.2 Human-Machine Collaborative Design

Although network pruning can result in pruned subnetworks that are more efficient and meet the original test accuracy, there lacks human intervention. As a result, the pruned network may not fit the task at hand. The human-machine collaborative design strategy combines the design requirements and constraints of the system with machine driven network design to create efficient neural networks suited for the task. Human-machine collaborative design strategy has shown success in designing highly compact deep neural networks well suited for various perception tasks, such as object detection, image classification and semantic segmentation [54, 55]. To achieve the required efficient neural networks, human-machine collaborative design comprises of two main design stages: i) principled network design prototyping, and ii) machine-driven design exploration [55].

Principled Network Design Prototyping

For network pruning, the base architecture depends on manual architecture selection strategies. The overall architecture of the final network produced by pruning techniques is the same as the base architecture. The only difference is that there are less nodes and connections. For human-machine collaborative design strategy, the network design prototyping stage is the initial design stage, where an initial network design prototype is created (denoted as φ) based on human-driven design principles, such as the designs of an encoder-decoder architecture and skip connections described in Sections 2.2.1 and 2.2.2, respectively, to guide the machine-driven design exploration stage. The final macroarchitecture and microarchitecture designs of the network are left to the machine-driven design exploration stage to design in an automatic manner.

Machine-driven design exploration

The machine-driven design exploration stage takes in the given data, initial network design prototype φ, and human-specified design requirements and constraints, which are designed specifically around edge scenarios with limited computational and memory capabilities.

Using the initial network design prototype φ described in the previous section, as well as human specified design requirements, a machine-driven design exploration is leveraged

in the form of generative synthesis [55] to determine macroarchitecture and microarchitecture designs for depth estimation on edge devices. The process of generative synthesis is capable of determining the optimal network macroarchitecture and microarchitecture design that satisfy the human-specified constraints. This is achieved by learning generative machines that can generate deep neural networks that meet the specified constraints. To learn the optimal generator, generative synthesis is formulated as a constrained optimization problem, defined in Equation 2.5, where given a set of seeds $\mathcal{S}$, a generator $\mathcal{G}$ can generate networks $\{\mathcal{N}_s | s \in \mathcal{S}\}$ that maximize a universal performance function $\mathcal{U}$, while also satisfying constraints defined in an indicator function $1_r(\cdot)$.

$$\mathcal{G} = \max_{\mathcal{G}} \mathcal{U}(\mathcal{G}(s)) \text{ subject to } 1_r(\mathcal{G}(s)) = 1, \forall s \in \mathcal{S} \tag{2.5}$$

The generative synthesis process is guided by both the initial prototype φ and human-specified constraints. To guide the process towards learning generative machines that generate highly efficient and compact depth estimation networks for edge devices, an indicator function $1_r(\cdot)$ is configured so that the generated networks are within the human-specified constraints [55].

Chapter 3

System Overview

A human-centered design strategy is taken in the design of OLIV, an AI-powered assistant that assists the visually impaired in locating displaced items. OLIV is comprised of three main components: speech recognition, custom-designed visual scene understanding, and synthesis, as shown in Fig. 3.1. Speech recognition allows individuals to independently query and interact with the system, increasing their level of independence. Visual scene understanding efficiently performs on-device object detection and depth estimation. By performing on-device visual scene understanding, personal data does not need to be sent to cloud-based computing resources and can stay on the user's local device, which gives the user control over who has access to their data. Since the personal data is not sent or stored to a cloud-based resources, privacy concerns over malicious attacks or unauthorized use of personal data can be avoided. Synthesis then combines the detected objects along with their locations and depths with the user's intent to construct a verbal semantic description that is verbally conveyed via speech synthesis.

3.1 Speech Recognition

The speech recognition of OLIV is responsible for interpreting the verbal query from the user as well as providing a verbal response to the user's query to inform the user with directions on where the displaced item is. To realize the capabilities needed for the speech module, off-the-shelf commercial smart assistant solutions, such as Microsoft Cortana, are leveraged as the interface and feedback to the user.

For this thesis, Microsoft Cortana, a smart voice assistant, is chosen for not only its state-of-the-art capabilities in speech recognition and speech generation, but also based

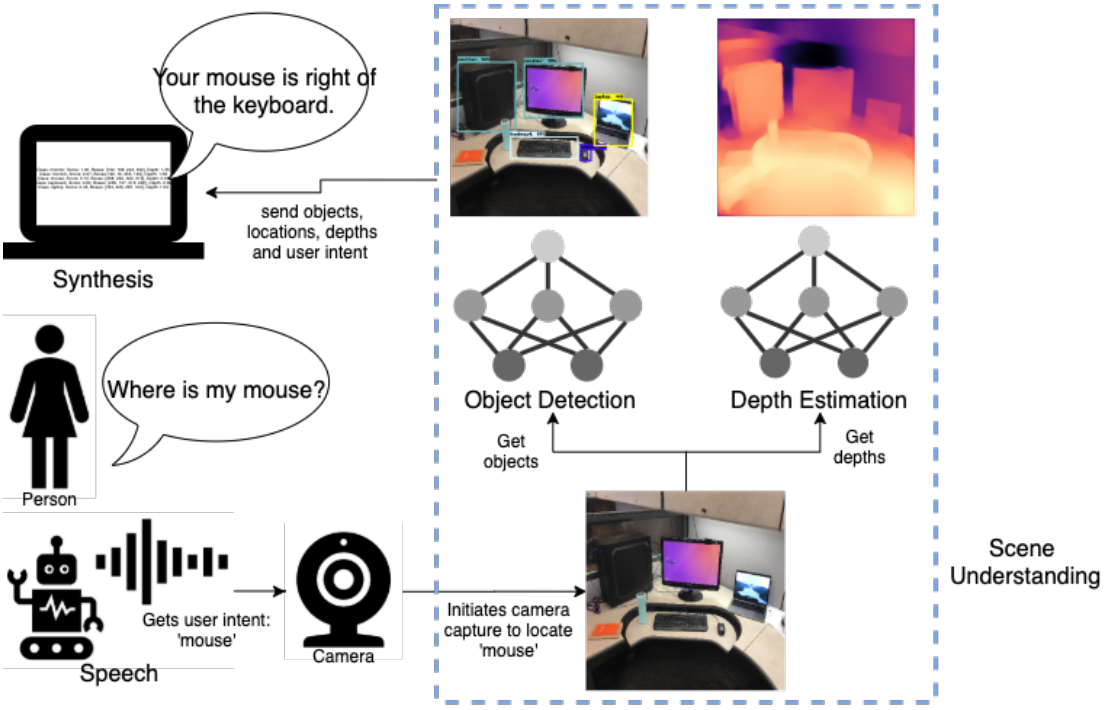


Figure 3.1: Overview of OLIV. The speech recognition algorithm takes in captured verbal speech to understand the user's query. The scene understanding algorithms and performs object detection and depth estimation on the captured scene. The detected objects and their associated depths are then used to synthesize a coherent response for the user, which is then verbally conveyed via speech synthesis.

on widespread adoption. To query OLIV for the location of an object, the user will ask Microsoft Cortana where the object is. Using speech recognition, Microsoft Cortana converts speech to text to get the object the user wants to locate. Microsoft Cortana will then use speech generation to communicate to the user a description of where the object is located.

By utilizing off-the-shelf smart assistant solutions, such as the use of Microsoft Cortana for OLIV, the devices are not only used for this particular system but also what users currently use them for. Home assistants are also affordable and accessible, making these devices easy to obtain.

3.2 On-device Scene Understanding for Privacy

An important component of OLIV is scene understanding, which uses object detection and depth estimation to build up a representation of the surrounding 3D scene. State-of-the art deep neural networks for the two tasks have been shown to achieve superior performance, but require high computation and memory, making them infeasible for on-device operation. Alternatively, cloud-based operation leads to privacy concerns.

To address these challenges, both manual micro-architecture design exploration and human-machine collaborative design strategy are taken to generate compact neural network architectures, resulting in private, on-device scene understanding. More specifically, for object detection, the problem of finding an optimal network architecture design is formulated as a numerical optimization problem, with the objective function being the NetScore [55], which quantifies the balance between accuracy, size, and speed, constrained by human-specified requirements for on-device scenarios. The result of these design exploration strategies resulted in a highly compact single-shot network architecture for object detection, as explained in Section 4, and a highly compact densely-connected encoder-decoder network architecture for monocular depth estimation, as described in Section 5.

In addition, by incorporating both on-device object detection and depth estimation, the stored images can be more anonymous. In addition to the fear of personal information being exploited, the use of cameras are also found to be obtrusive. By saving depth maps, along with object class labels, detailed information, such as a person and personal information, can be anonymous. Depth estimation and object detection would occur on the local device, however, if these images need to be sent to through the cloud to family members, caretakers or medical professionals, the anonymized images will protect personal details in case of a malicious attack. For instance, in Fig. 3.2, the person and object details on the monitors are replaced by depth map representations.

3.3 Synthesis

The purpose of synthesis is to interact with speech recognition and scene understanding components to understand user intent and construct a semantic description based on intent and scene understanding results so that a verbal response can be provided for the user. Synthesis achieves these connections using three main functions. The first function is to determine what the object of interest is based on the interpreted query passed in by speech recognition. The next function is to initiate image capture so scene understanding can

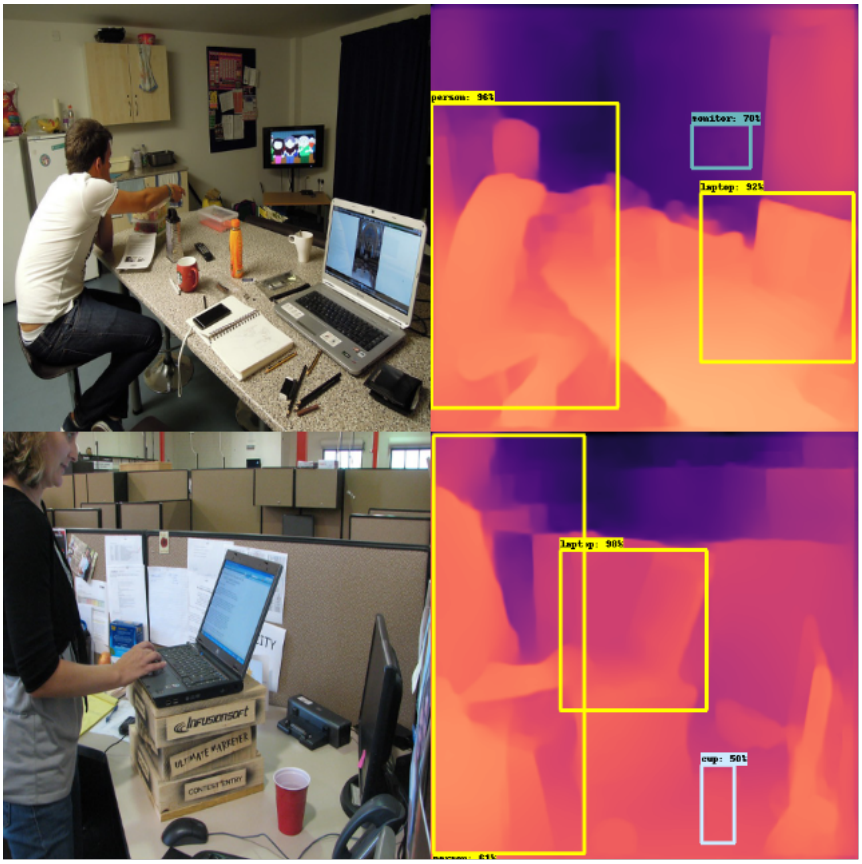


Figure 3.2: Data anonymization. (Left) rgb image from MSCOCO dataset [32], (right) anonymous image with detected objects. For both these images, the person, as well as private information on the laptop screens can be protected.

determine what objects are in the scene and their locations. The third function is to build a semantic description response based on the objects in the scene, the locations and the user intent for what he or she is looking for.

The first and last function of synthesis uses a web-hook, which is a way for an application to deliver data to other applications, to relay the request from the speech module and the response to the speech module, respectively. Once the request is received, the sentence is parsed for a word that matches one of the objects in the dataset or a synonym of it by following the structure for WordNet, which is a large lexical database of English nouns, verbs, adjectives and adverbs grouped into sets of cognitive synonyms, also known as synsets [36]. For instance, if a user asks for a "bottle", this will correspond to the "water

bottle“ class in the dataset. For the second function of requesting the camera to capture the scene, the camera is connected to the system as a video source so only a command is needed to activate the camera. Once the type and locations of all the objects are identified by scene understanding, synthesis then identifies the target object of interest and calculates which object is closest based on the bounding box location and depth of the object, known as the reference object, as well as the relative position to the reference object. For instance, in Fig. 3.3 the target object is “mouse“ and the reference object is “keyboard“. Since the mouse is around the same depth as the keyboard, the relative location of “right“ is used to explain the position of the mouse.

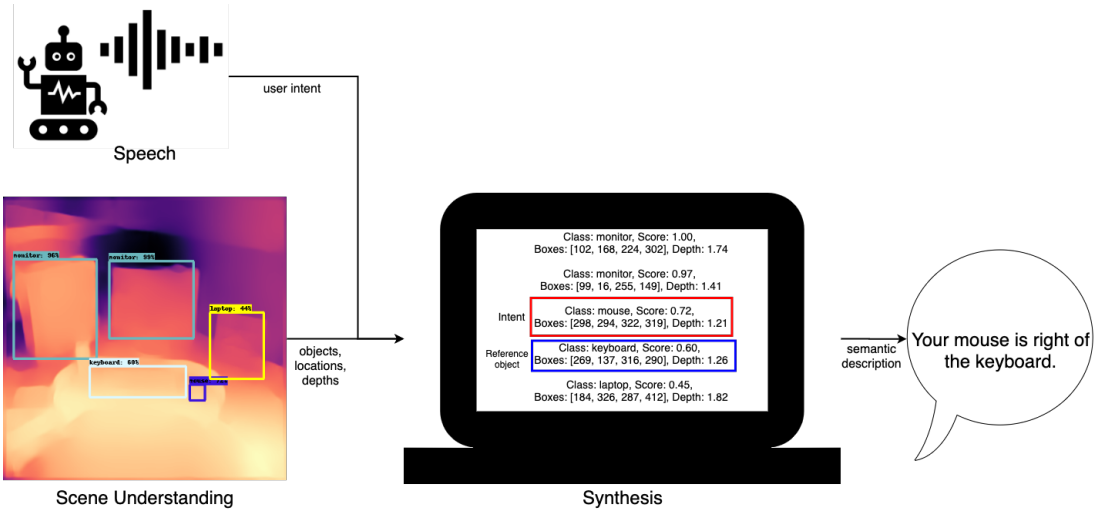


Figure 3.3: Synthesis process for a sample workplace scenario. Synthesis takes input from speech recognition and scene understanding, then builds a description of the relative position of the target object to the closest reference object.

3.4 Summary

A novel AI-powered assistant, OLIV, is proposed to aid individuals with visual impairment in locating displaced objects. To design the OLIV system, a human-centered design strategy was taken to ensure that personal data remains on the inference device. OLIV consists of three main components: speech recognition, custom-designed visual scene understanding and synthesis. First, speech recognition utilizes off-the-self smart assistant solutions,

such as home assistants, to understand and communicate with the user. Then, visual scene understanding combines object detection and monocular depth estimation to gain an understanding of the surrounding 3D indoor environment. Synthesis finally combines the user's request with the gathered 3D scene to produce a semantic description based on locations of other objects in the scene for the user.

An important component of OLIV is data privacy. In order for both the object detection network and monocular depth estimation network to run efficiently on a local device, two different architectural design exploration methods are employed. The methods and results to design the highly compact object detection and monocular depth estimation network architectures are described in Section 4 and 5, respectively.

Chapter 4

Highly Compact Single-Shot Network Architecture for Object Detection

In this chapter, the process of micro-architecture design exploration to design a highly compact single-shot object detection network architecture for OLIV is described. The micro-architecture design exploration and architectural design optimization to quantitatively find the best performance tradeoff are presented in Section 4.2 and 4.3, respectively. Experimental setup, including the dataset used, is outlined in Section 4.4. Both visual and quantitative experimental results are detailed in Section 4.5.

4.1 Problem Formulation

Object detection plays an integral role in OLIV, however, many object detection neural networks are large and require powerful graphical processing units (GPUs) to increase modeling capabilities. The increasing complexity of these deep neural networks is one of the obstacles to widespread deployment on edge devices, where computational power and memory capacity are significantly lower than parallel computing on GPUs. Edge devices provide both affordability and autonomy, which are mutually exclusive for systems that require GPUs. If the device contains a local GPU to protect autonomy over personal data, affordability decreases. As an alternative, cloud services provide a more affordable option, however, privacy concerns arise over who has access to personal data.

To address these challenges, this chapter investigates a network architecture exploration approach to creating an object detection network specifically for OLIV, an AI-powered

assistant for object localization for impaired vision, shown in Figure 3.1. By enabling object detection to run on edge devices and mobile devices, cost and privacy concerns are reduced. This means that the object detection network must be able to perform on-device object detection with low computational power in resource constrained environments.

4.2 Micro-architecture Design Exploration

Object detection contains two parts, the meta-architecture and feature extractor. To design the appropriate micro-architecture, the meta-architecture and feature extractor must meet the design requirements for OLIV.

For object detection to be applied in an indoor environment to assist individuals with visual impairment, fast inference time on a local device is required in order to reduce privacy concerns and costs, as well as to provide a more natural response time. To meet these requirements, single shot detector (SSD) is chosen as the meta-architecture, as described in Section 2.1.1.

For the feature extractor, MobileNetV2 is found to offer the best tradeoff between accuracy and speed for image classification. MobilenetV2 is a neural network architecture for image classification that is specifically tailored for mobile and resource constrained environments [44], as described in Section 2.1.2.

Taking into account the specialized application of OLIV for indoor environments, the design of the small and fast architecture of MobileNetV2-SSD is taken one step further. Given the reduced number of target classes for indoor environments, an exploration of hyper-parameters is conducted to find the hyper-parameters that offer the best trade-offs between performance, speed and size for this specific application.

MobileNetV2 includes two additional hyper-parameters, width and resolution multiplier, as described in Section 2.1.2, to the MobileNetV2 base architecture in Table 4.1. The width multiplier, α, reduces the width of the network uniformly at each layer by multiplying the number of input channels and output channels in the MobileNetV2 base architecture by α. For instance, with $\alpha = 0.75$, the first three output channels in Table 4.1 will be 24, 12 and 18. The resolution multiplier, ρ, reduces the representation by multiplying each feature map by ρ [44]. If $\rho = 0.85$, the MobileNetV2 base architecture input resolution of 224×224 will by multiplied by 0.85 for an adjusted input resolution of 192×192. Although reducing the width and resolution makes the model smaller and faster, there is a cost to the accuracy, which is explored in Section 4.5.

Table 4.1: MobileNetV2 base architecture adapted from Sandler et al. [44]. Each line contains 1 or more identical layers repeated n times. k represents the number of classes for classification.

Input	Type of Module	Output Channels	Times Repeated
224x224x3	conv2d 3x3	32	1
112x112x32	inverted residual block	16	1
112x112x16	inverted residual block	24	2
56x56x24	inverted residual block	32	3
28x28x32	inverted residual block	64	4
14x14x64	inverted residual block	96	3
14x14x96	inverted residual block	160	3
7x7x160	inverted residual block	320	1
7x7x320	conv2d 1x1	1280	1
7x7x1280	avgpool 7x7	-	1
1x1x1280	conv2d 1x1	k	-

In order to select the appropriate ranges of the input resolutions and width multipliers, the requirements of this specific application is considered. For the application to remain affordable, the quality of the camera becomes a constraint. Taking this into account, practical input resolutions of 300, 224, 192 and 160 pixels are used, which correspond to resolution multipliers of $\rho = 1.34, 1.0, 0.85, 0.71$, respectively. To analyze the effects of the input resolution, the width multiplier is kept constant at 1.0. Not only does the quality of the camera affect cost, but also the amount of memory used. To keep the size of the network small, width multipliers from 1.4 to 0.35 are explored while keeping the input resolution constant at 224. Only width multipliers in the range from 1.4 to 0.35 are explored based on the MobileNetV2 performance curve, which showed accuracy reaching a plateau as the width multiplier increased past 1.4 and significant reduction in accuracy as the width multiplier decreased past 0.35 [44]. The exploration consisted of 9 possible models, as shown in Table 4.2, along with corresponding network architecture complexity of the deep neural network, which were calculated using the Tensorflow profiler tool [49].

By varying one hyperparameter and keeping the other one constant and vice versa, interpolation can be used to compute the remaining pairings. This allows for a complete micro-architecture exploration of models that satisfy the requirements for this application.

Table 4.2: MobileNetV2-SSD hyper-parameter values used for the experiments and network architecture complexity (number of parameters).

Model Name	Width Multiplier	Input Resolution (pixels)	Parameters (M)
MobileNetV2-SSD-0.35_224	0.35	224	1.15
MobileNetV2-SSD-0.5_224	0.5	224	1.82
MobileNetV2-SSD-0.75_224	0.75	224	3.13
MobileNetV2-SSD-1.0_160	1.0	160	4.71
MobileNetV2-SSD-1.0_192	1.0	192	4.71
MobileNetV2-SSD-1.0_224	1.0	224	4.71
MobileNetV2-SSD-1.0_300	1.0	300	4.71
MobileNetV2-SSD-1.3_224	1.3	224	6.99
MobileNetV2-SSD-1.4_224	1.4	224	7.83

4.3 Architectural Design Optimization

The network architecture exploration approach is formulated as a numerical optimization problem, where the goal is to find the set of width and resolution hyperparameter pairs, defined as $\underline{\theta} = \{\alpha, \rho\}$, that maximize a modified NetScore. NetScore is a quantitative assessment of the balance between accuracy, computational complexity and network architecture complexity of a deep neural network [53]. For this study, the number of multiply-accumulate operations, which represent the computational complexity, is replaced with CPU running time, as CPUs are on edge devices and mobile devices. In terms of on-device object detection, there is a limited amount of parallel computation, thus, speed is found to be more representative. The modified NetScore function is defined as:

$$\Omega(\mathcal{N}(\underline{\theta})) = 20 \log \left(\frac{a(\mathcal{N}(\underline{\theta}))^{\kappa}}{p(\mathcal{N}(\underline{\theta}))^{\beta} r(\mathcal{N}(\underline{\theta}))^{\gamma}} \right) \tag{4.1}$$

where $a(\mathcal{N}(\underline{\theta}))$ is the accuracy of the network, $p(\mathcal{N}(\underline{\theta}))$ is the number of parameters in the network in millions, $r(\mathcal{N}(\underline{\theta}))$ is the CPU running time in seconds, and κ, β, γ control the the influence of accuracy, architectural complexity and computational complexity, respectively. Since the application requires the network to run on a local device, the size is constrained, as mobile devices usually have at most 8 GB of RAM [22]. As well, in order to offer a natural conversational system, the inference on CPU should approach real-time. While architectural and computational complexity are important for this application, accuracy remains as the most important metric in the objective function since accuracy shows the potential usefulness and benefits to the user's current lifestyle. The weighting for accuracy is set to $\kappa = 1$. For this application, speed is also important for a smooth flow and is set to

$\beta = 0.45$. Since MobileNetV2-SSD is already a small network with a maximum of 7.83M parameters in this study, the architectural complexity is set to $\gamma = 0.2$.

The formal objective function used to determine the most optimal hyper-parameters for this application is defined as:

$$\hat{\underline{\theta}} = \underset{\underline{\theta}}{\operatorname{argmax}}\, \Omega(\mathcal{N}(\underline{\theta})) \tag{4.2}$$

where $\hat{\underline{\theta}}$ is the width and resolution hyperparameter pair that maximizes the modified NetScore, $\Omega(\mathcal{N}(\underline{\theta}))$. To find the most optimal hyperparameter pair for OLIV, experiments on the different model configuration in Table 4.2 are conducted. Interpolation is then used to determine the remaining hyperparameter pairs between width multipliers from 1.4 to 0.35 and resolution multiplier from 1.34 to 0.71.

4.4 Experimental Setup

4.4.1 Dataset

Since most daily tasks are performed in a indoor setting [3], the objects used in this paper are ones that could be found indoors. For this study, a subset of the MSCOCO dataset that contains indoor objects are compiled to form the MSCOCO-OLIV dataset. Compared to ImageNet and PASCAL VOC, MSCOCO contains objects in their natural context, allowing for more generalization in the real-world and more realistic to what is seen in an indoor environment. In addition to non-iconic images, there are on average 7.7 instances per image in COCO, whereas ImageNet and PASCAL VOC have less than 3 instances per image [32].

Although COCO has more instances per image, the object sizes are also smaller [32]. In the specific application environment for OLIV, the camera is placed closer to the objects, allowing the objects to be bigger. In order to be more representative of the application environment, the evaluation of the models on MSCOCO-OLIV is performed on large objects.

To obtain the MSCOCO-OLIV dataset, a subset of the labeled COCO2017 train and val images that contain indoor objects were used, resulting in 21 classes. The 21 target classes are reported in Table 4.3. Of the subset, 66674 images were used for training, 8889 images were used for validation and 13334 images were used for testing.

4.4.2 Training

Each model's feature extractor architecture is initialized with the pretrained weights from the corresponding MobileNetV2 classification model, except MobileNetV2-SSD-1.0_300, which used mobilenet_v2_1.0_224 since there are no pretrained MobileNetV2 configurations with a width multiplier of 1.0 and input resolution of 300. All the MobileNetV2 classification models were trained on the ImageNet dataset and obtained from [45]. These pretrained classification models are the feature extractors for the models in Table 4.2. The models in Table 4.2 were then further trained end-to-end on MSCOCO-OLIV for 800000 steps using an initial learning rate of 0.004, decaying by a factors of 0.95 every 200720 steps.

4.4.3 Performance Evaluation Metrics

Each MobileNetV2-SSD network is evaluated using mean average precision (mAP) at intersection over unions (IoUs) from 0.5 to 0.95 with a step size of 0.05, which is the measure for object detection [32]. To evaluate the real-world performance of the MobileNetV2-SSD networks, inference speed on a 2.3GHz Dual-Core Intel Core i5 CPU is performed.

Intersection over union (IoU), shown in Eq. 4.3, is used to evaluated the accuracy of the predicted bounding boxes. IoU is calculated as the area of intersection of the ground truth bounding box (A_G) and the predicted bounding box (A_P) over the area of union between the ground truth bounding box and predicted bounding box.

$$IoU = \frac{A_G \cap A_P}{A_G \cup A_P} \tag{4.3}$$

In order to compute mAP for the object detection results on the MSCOCO-OLIV dataset, average precision (AP) is first computed across IoUs from 0.5 to 0.95 with step size of 0.05 for each object class. AP is the number of true positive predictions (TP) over the number of true positive and false positive predictions (FP) for each class, as shown in Eq. 4.4.

$$AP = \frac{TP}{TP + FP} \tag{4.4}$$

The mean average precision (mAP) is then computed as the average precision for all classes averaged over all the classes. mAP is computed using Eq. 4.5 below:

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{4.5}$$

where Q is the number of classes in a dataset and AP is the average precision for a given class, q.

Inference speed is measured by the CPU time of how long one pass through the model takes. Inference starts when the image is loaded into the model and stops at the output of all detections after non-maximum suppression.

4.5 Experimental Results

To study the utility of MobileNetV2-SSD modifications on the width and resolution multiplier, the overall and class-wise mAP for MSCOCO-OLIV and average CPU times (on a 2.3GHz Intel Core i5 processor) are examined on the test dataset. In addition, class-wise average precision results and trends between the different model variations are also analyzed.

4.5.1 Accuracy vs. Speed

The performance results with respect to overall mAP and average CPU time of 9 different SSD-MobileNetV2 models are shown in Figure 4.1. Based on the overall mAP of 22.1% achieved by SSD-MobileNetV2-1.0_300 on the full MSCOCO dataset [33], MobileNetV2-SSD-1.0_300, MobileNetV2-SSD-1.3_224 and MobileNetV2-SSD-1.4_224 show the most promise on MSCOCO-OLIV. However, the most optimal object detection model for OLIV depends on a balance between accuracy and speed as described in Section 4.3.

A number of observations can be made. First, as CPU time increases, there are diminishing returns to the improvement of accuracy, as shown by the trend line. Therefore, for indoor environments where there are a less variety of objects, the largest model is not necessarily the best choice for applications where speed and size are a constraint. Second, MobileNetV2-SSD-1.0_160 and MobileNetV2-SSD-0.5_224, as well as MobileNetV2-SSD-1.0_192 and MobileNetV2-SSD-0.75_224, achieve similar run-time and mAP for each pair. However, in each pair, the number of parameters is less for the smaller width multiplier, even though the models with a width multiplier of 1.0 have a smaller resolution. This result shows that the input resolution has to be sufficiently large enough to resolve the objects in the scene. As well, this result also indicates that adjusting the width multiplier achieves the better trade-off between accuracy, speed and size. This is because changing the input resolution does not affect the number of parameters. Based on performance

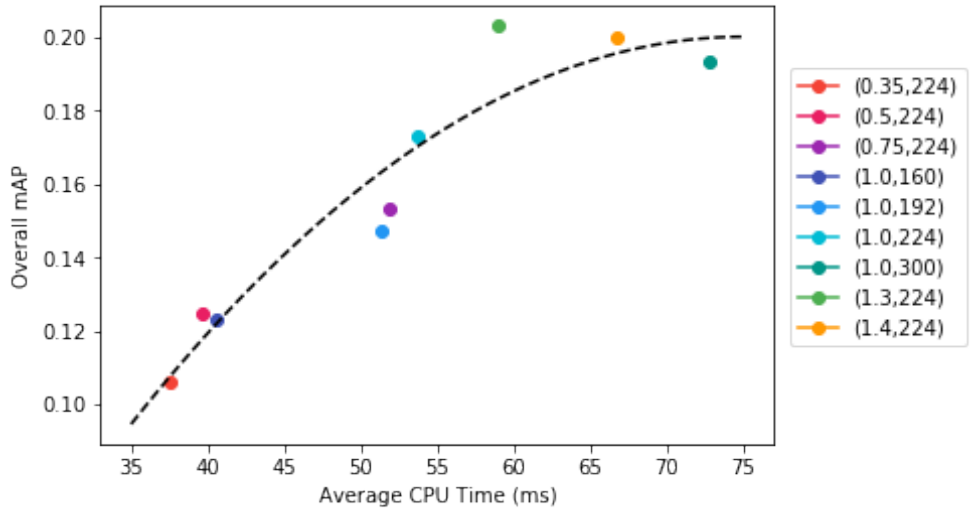


Figure 4.1: Accuracy vs time: how the different model configurations in Table 4.2 affect average precision and CPU time.

results with respect of overall mAP and average CPU time, the choice of width multipliers for MobileNetV2-SSD models have more influence over accuracy, speed and size than resolution multipliers.

4.5.2 The Effect of Width and Resolution Multiplier

Figure 4.1 shows the relationship between overall mAP and run-time. To offer a more in-depth insight into the effects of the the width multiplier and input resolution on overall mAP and CPU running time, results for the hyperparameter pairs that were not ran during the experiment are interpolated. The interpolated results are visualized as a heatmap in Figure 4.2.

Based on the results, both the width multiplier and input resolution have a fairly linear relationship with speed, as shown in the right plot in Figure 4.2. On the contrary, the width multiplier and input resolution have a non-linear relationship with mAP, as shown in the left plot. Referring to the overall results in Table 4.3, as the width multiplier increases from 0.35 by 42.8% to 0.5, there is a 17.9% increase in mAP. From 0.75 to 1.0, mAP increases by 13.1% for 33% increase in width. As well, from 1.0 to 1.3, mAP increases by 17.3% for the 30% increase in width. However, from 1.3 to 1.4, there is an decrease of 1.48% in mAP for 7.7% increase in width.

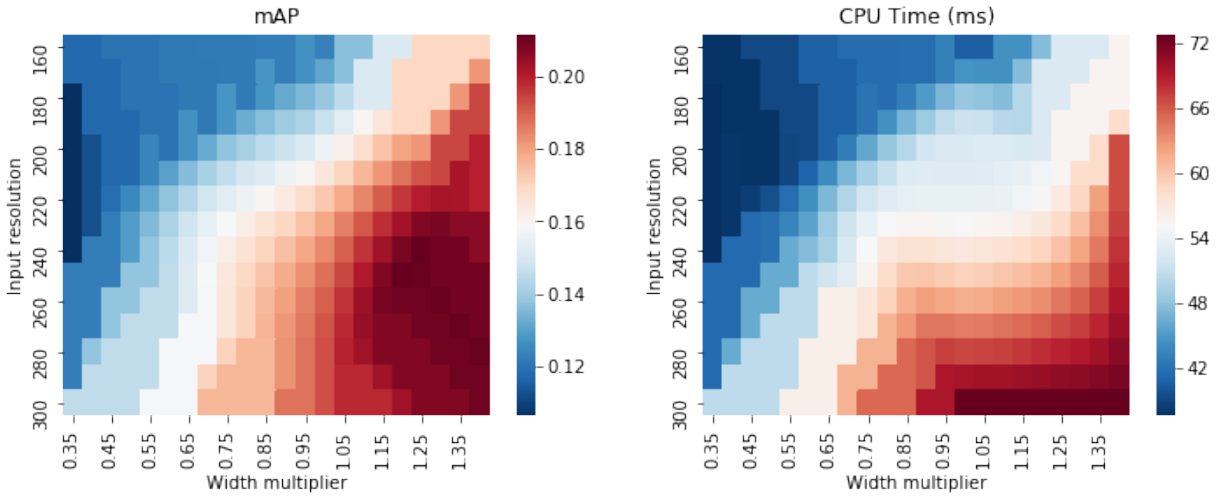


Figure 4.2: Effect of width and resolution multiplier on accuracy (left) and speed (right)

Similarly, as the input resolution increases from 160 to 300, the mAP first increases by 19.5% for 20% increase in resolution, then 17.6% for 16.7% increase in resolution, and finally 11.5% for 33.9% increase in resolution. Although there was a significant increase in resolution from 224 to 300, the increase in mAP is insignificant compared to the rest.

These results show that in general, as the width multiplier and input resolution increase, the mAP and CPU running time also increase. However, there is a certain point where there are diminishing returns in the increase of mAP. Based on the mAP heatmap, there is little increase in mAP after the width multiplier reaches 1.15 and input resolution reaches 220. However, in the CPU time heatmap, run-time continues to increase linearly as the width multiplier and input resolution keeps increasing from 1.15 and 220. Although the accuracy of OLIV is important, as described in Section 4.3, the amount of additional time required to run a model needs to be considered to maintain a balanced tradeoff between accuracy and speed to ensure the on-device computation is providing users with a fast and accurate description of where the object is.

4.5.3 Class-wise Performance Analysis

All 21 class precisions for each model are shown in Table 4.3. For all the different width multipliers and input resolutions, there is a large variation between classes. This can be expected since the number of instances for each class varies in the MSCOCO-OLIV dataset.

For instance, the person class has almost 10 times more instances than the fork class, which is shown by a decrease in average precision from person class to fork class in Table 4.3.

Observing Table 4.3, the average precision remains relatively consistent for width multipliers of 1.4 and 1.3. There is a decrease when the width goes down to 1.0, but there is only a slight decrease when the width decreases from 1.0 to 0.75. The width multipliers of 0.5 and 0.35 show another drop in average precision for all the classes. Between 1.4 and 1.3, as well as between 1.0 and 0.75, the smaller width multiplier is preferred because there is a small decrease in accuracy but a linear decrease in CPU running time, as shown in Figure 4.2.

Table 4.3: Class-wise average precision results by increasing number of parameters

	Models								
Classes	**0.35, 224**	**0.5, 224**	**0.75, 224**	**1.0, 160**	**1.0, 192**	**1.0, 224**	**1.0, 300**	**1.3, 224**	**1.4, 224**
Overall	0.106	0.125	0.153	0.123	0.147	0.173	0.193	0.203	0.2
Bottle	0.055	0.063	0.105	0.058	0.092	0.106	0.175	0.156	0.192
Person	0.337	0.353	0.396	0.339	0.366	0.417	0.443	0.439	0.445
Backpack	0.002	0.011	0.015	0.01	0.005	0.014	0.034	0.042	0.048
Handbag	0	0	0.007	0.009	0.003	0.021	0.013	0.016	0.016
Umbrella	0.076	0.095	0.137	0.119	0.126	0.146	0.178	0.199	0.177
Bowl	0.169	0.19	0.203	0.191	0.205	0.22	0.233	0.284	0.246
Cup	0.122	0.138	0.203	0.129	0.188	0.251	0.292	0.31	0.266
Clock	0.306	0.384	0.423	0.318	0.414	0.434	0.488	0.456	0.467
Knife	0.009	0.007	0.042	0.022	0.033	0.043	0.063	0.075	0.058
Spoon	0	0.009	0.011	0.004	0.014	0.025	0.031	0.035	0.043
Fork	0.007	0.022	0.067	0.048	0.056	0.089	0.079	0.125	0.093
Chair	0.023	0.037	0.054	0.038	0.056	0.074	0.097	0.097	0.118
Potted Plant	0.055	0.081	0.096	0.085	0.081	0.126	0.154	0.141	0.152
Dining Table	0.224	0.223	0.246	0.21	0.22	0.238	0.223	0.268	0.247
Keyboard	0.154	0.176	0.173	0.194	0.236	0.25	0.234	0.255	0.26
Mouse	0.127	0.202	0.235	0.16	0.206	0.223	0.293	0.316	0.376
Laptop	0.216	0.242	0.294	0.22	0.313	0.347	0.377	0.356	0.37
Cellphone	0.089	0.094	0.089	0.13	0.124	0.138	0.175	0.179	0.156
Scissors	0.006	0.028	0.067	0.03	0.03	0.094	0.112	0.105	0.09
Book	0.012	0.019	0.042	0.017	0.022	0.04	0.039	0.043	0.041
Monitor	0.236	0.249	0.311	0.25	0.3	0.348	0.314	0.374	0.349

Likewise, as the input resolution decreases, the class average precision also decrease. Unlike how average precisions are similar for width multipliers of 1.4 and 1.3, as well as 1.0 and 0.75, the average precisions in each class decrease linearly with input resolution.

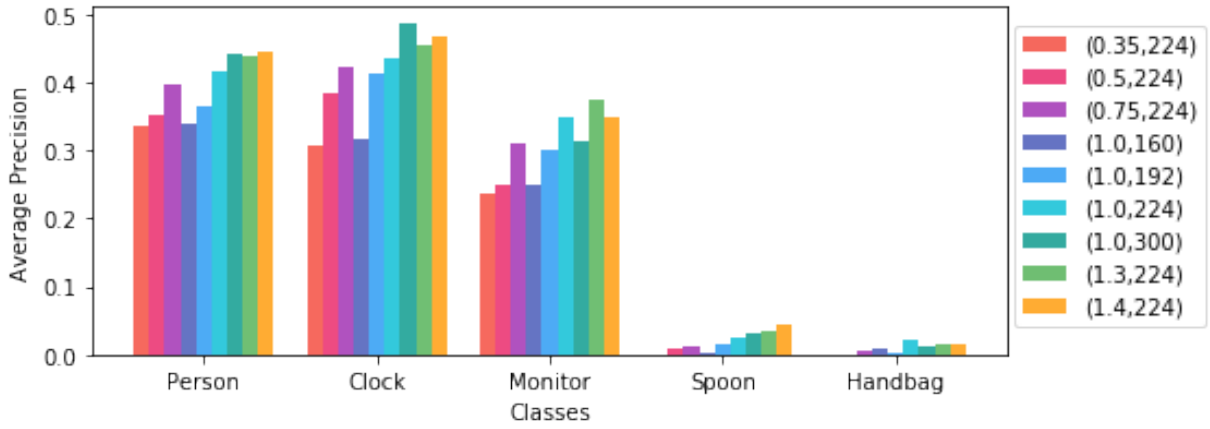


Figure 4.3: Class-wise trends for top 3 and bottom 2 classes as width and input resolution varies.

To further analyze the class-wise relationships, Figure 4.3 shows a visualization of how the different models, ordered by increasing number of parameters, affect average precision of the top 3 and bottom 2 objects. All classes show an increase in average precision as the width multiplier increases with constant input resolution. From visual inspection in Figure 4.3, the largest jump in precision is seen between width multipliers of 0.75 and 0.5 for person, monitor and handbag and between 0.5 and 0.35 for clock and spoon.

Likewise, there is also an increase in average precision as the input resolution increases while the width multiplier stays constant. Unlike the trend of the width multiplier, where average precision increases with width for every class, as the input resolution increases from 224 to 300, not all the average precisions increase, as shown for monitor and handbag. For instance, in the left most image in Fig. 4.4, the handbag covers a significant portion of the image compared to the other objects, yet remains undetected by MobileNetV2-SSD-1.3_224, which is the model that achieves the highest MAP. This suggests that adjusting the width multiplier to learn more features is more important than increasing the resolution.

In terms of number of parameters, although MobileNetV2-SSD-1.0_160 and MobileNetV2-SSD-1.0_192 have more parameters than MobileNetV2-SSD-0.75_224, MobileNetV2-SSD-0.75_224 achieves higher average precision for most of the classes. This shows that selecting

Figure 4.4: Sample detection results on MSCOCO-OLIV using MobileNetV2-SSD-1.13_224. All objects in Table 4.2 could be detected, however, left) the spoon is not able to be detected, middle) the book and small objects like the mouse are missed, right) even though the handbag is the largest object, it is not detected.

hyperparameter pairs is important and can result in a smaller MobileNetV2-SSD model achieving better results than a larger model.

4.5.4 Architecture Exploration Analysis: Balance Between Accuracy, Speed and Size

Although the previous analysis results offer insight on the effects of width and resolution hyper-parameters, those results do not offer a quantitative answer as to which model offers the best trade-offs for the case of assistive devices for the visually impaired. As such, as mentioned in Section 4.3, we employ an architectural design optimization strategy to identify the model with the best balance in terms of accuracy, speed and size for indoor object detection on an edge device. To take a deeper look into this architecture design optimization process, Figure 4.5 illustrates how the modifed NetScore objective function, defined in Equation 4.2, changes depending on the width multiplier and input resolution.

Based on Figure 4.5, MobileNetV2-SSD-1.3_224 achieves the highest modified NetScore of 68.7, indicating that this particular model offers the best balance between accuracy, speed and size for this specific application. From Figure 4.1, MobileNetV2-SSD-1.3_224 achieved the highest overall mAP score, while achieving faster CPU running time than MobileNetV2-SSD-1.4_224 and MobileNetV2-SSD-1.0_300. This result is also supported by Figure 4.2, which illustrated that there are smaller increases in mAP than speed as the

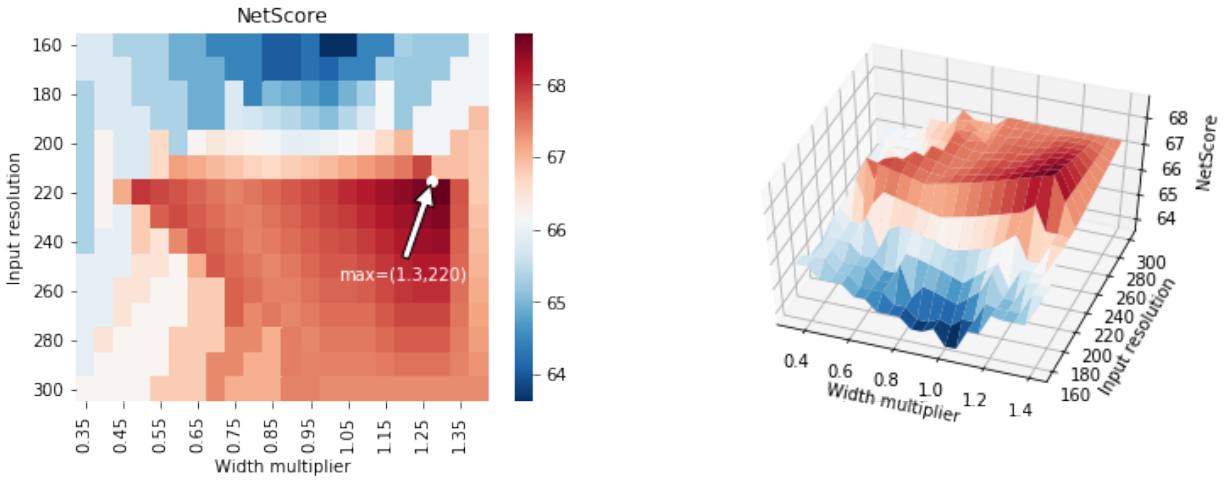


Figure 4.5: Modified NetScore results shown as a heatmap (left) and surface plot (right). The maximum modified Netscore is achieved using with hyperparameters $\alpha = 1.3$ and $\rho = 224$, indicated by the white arrow.

width multiplier passes 1.15 and input resolution reaches 220. By conducting an architectural design optimization strategy, the most optimal model with the desired tradeoffs can be determined.

4.6 Summary

In this section, a micro-architecture exploration approach to create an object detection network that can run on-device specifically for OLIV is investigated. Based on the requirements to increase affordability and autonomy, as well as usefulness, the optimal network needs to offer a balanced trade-off between accuracy, speed and size. To quantify the balance required for this specific application, the network architecture exploration approach is formulated as a numerical optimization problem, where the goal is to find a set of hyperparameter pair, which in this case is the width and resolution multiplier, that maximizes the modified NetScore. Based on quantitative architecture exploration analysis, the model with the optimal trade-off between accuracy, speed, and size for this application is the model that achieves the highest NetScore, which in this case is MobileNetV2-SSD-1.3-224. This result is also supported by the experimental results as the width and resolution multiplier increase. By taking a micro-architecture exploration approach, a well-suited compact

object detection model that offers a balanced trade-off between accuracy, speed and size is found for OLIV.

Although the hyper-parameter combination with width multiplier of 1.3 and input resolution of 224 for MobileNetV2-SSD is the best model based on the objective function given in Equation 4.2, a different modified NetScore, as well as different κ, β and γ values will result in different model choices. For this application, accuracy, speed and size were weighted to offer a reliable, secure and low-cost solution. This approach can also be applied to other assistive devices to offer the user a cost-efficient and secure solution.

In addition to selecting the best feature extractor and meta-architecture for an object detection network given an application, micro-architecture exploration approach manually explores different hyperparameter combinations for the selected object detection network. For OLIV, the micro-architecture exploration approach manually explored different width and resolution multiplier combinations and the resulting trade-off between accuracy, speed and size for the highly compact single-shot object detection network architecture. In the next chapter, both a human and machine driven design exploration approach is used to design a highly compact densely-connected encoder-decoder network architecture for monocular depth estimation.

Chapter 5

Highly Compact Densely-Connected Encoder-Decoder Network Architecture for Monocular Depth Estimation

In this chapter, a human-machine collaborative design strategy is taken to design a highly compact densely-connected encoder-decoder monocular depth estimation network architecture for OLIV. Section 5.2 provides a detailed description of the human-machine collaborative design strategy leveraged in this chapter. Experimental setup containing the dataset and performance evaluation metrics are described in Section 5.3. Section 5.4 provides a detailed description and discussion of interesting characteristics of the resulting architecture design. Section 5.5 presents and discusses the results from the quantitative and qualitative experiments conducted to study the efficacy of the designed network when compared to state-of-the-art depth estimation networks.

5.1 Problem Formulation

The task of estimating depth from 2D images, also known as monocular depth estimation, is crucial for OLIV. The depth information gained from the scene enables OLIV to have a better understanding of the object positions to give the user an enriched response and experience. However, current state of the art methods for monocular depth estimation

have focused on an encoder-decoder architecture with very large and powerful deep feature extractor macroarchitecture designs, such as VGG, ResNet and DenseNet [15, 2, 29], to learn deep features. In addition to possessing very high network architecture complexity, these current deep neural networks have high computation time and low energy efficiency, which makes them difficult to deploy on mobile and edge devices to address user's privacy concerns.

Taking inspiration from recent work in efficient object detection, where large feature extractor architectures are replaced with more efficient network architectures, such as MobileNetV2 [44], Wofk et al. [52] used smaller feature extractor architectures (e.g., ResNet-16 and MobileNet) in order to decrease the number of parameters and run-time necessary to operate on embedded devices. To further reduce the network size and inference runtime, network pruning was applied [56]. While the resulting depth estimation networks achieved significant improvements in terms of inference speed, the depth estimation performance was significantly lower and not comparable with current state of the art in indoor environments.

Taking a different direction than manual architecture selection strategies and network pruning strategies, human-machine collaborative design strategy has shown recent success in designing highly compact DCNNs by coupling principled network design prototyping and machine-driven design exploration based on human-specified design requirements and constraints [55]. In particular, such strategies have been demonstrated to be quite effective at designing efficient deep neural networks well suited for various perception tasks, such as object detection, image classification, and semantic segmentation [54, 55].

To design a highly compact architecture for monocular depth estimation, we explore a human-machine collaborative design strategy to design highly compact deep convolutional neural networks for the task of monocular depth estimation on the edge. More specifically, we leverage encoder-decoder design principles that were found to be effective in current state of the art monocular depth estimation to create DepthNet Nano, a highly compact network with highly customized module-level macroarchitecture and microarchitecture designs tailored specifically for OLIV.

5.2 Human Machine Collaborative Design Strategy

5.2.1 Principled Network Design Prototyping

For human-machine collaborative design strategy, the network design prototyping stage is the initial design stage, as described in Section 2.3.2, where an initial network design

prototype is created based on human-driven design principles to guide the machine-driven design exploration stage. To create DepthNet Nano, an initial network design prototype was constructed based on densely-connected encoder-decoder architecture design principles [2], which has been demonstrated to be quite successful in achieving high-resolution monocular depth estimation for indoor scenes.

A standout characteristic of the densely-connected encoder-decoder architecture is the leveraging of a large number of direct connections between not only encoder layers, but also between encoder and decoder layers. A detailed description is provided in Section 2.2. It is very important to note that the actual macroarchitecture and microarchitecture designs of the individual modules and layers in the final DepthNet Nano network architecture, as well as the number of network modules, are left for the machine-driven design exploration stage to decide in an automatic manner based on both human-specified design requirements and constraints catered to edge device scenarios with limited computational and memory capabilities.

5.2.2 Machine Driven Design Exploration

As detailed in Section 2.3.2, machine driven design exploration is leveraged in the form of generative synthesis process to determine macroarchitecture and microarchitecture designs. Generative synthesis process is guided by both the initial prototype φ and human-specified constraints. To guide the process towards learning generative machines that generate highly efficient and compact depth estimation networks suitable for OLIV, the indicator function $1_r(\cdot)$, described in Equation 2.5, was set up such that: i) δ_1 accuracy ≥ 0.81 on NYU Depth v2, which is the dataset used to train DepthNet Nano as detailed in Section 5.3.2, and ii) network architecture complexity $\leq$ 5M parameters. The δ_1 accuracy and network architecture complexity conditions in the indicator function $1_r(\cdot)$ are set for this case such that the δ_1 accuracy of the resulting DepthNet Nano network exceeds that of Laina et al. [29], a popular deep convolutional neural network for monocular depth estimation for NYU Depth v2, while having more than 8$\times$ fewer parameters.

5.3 Experimental Setup

5.3.1 Implementation Details

The proposed DepthNet Nano was implemented using the TensorFlow open source platform for machine learning. The training scheme leveraged in this study was that outlined in [2],

as [2] was able to achieve state of the art results for indoor monocular depth estimation. The ADAM optimizer [24] was leveraged with a learning rate of 0.00005 and parameter values $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

5.3.2 Dataset

Since OLIV is designed for indoor environments, DepthNet Nano is trained on the NYU-Depth V2 dataset. The NYU-Depth V2 dataset provides 2D images and corresponding depth maps captured using Microsoft Kinect of 464 unique indoor scenes [46]. DepthNet Nano is trained on 50,000 images, which is a subset of the NYU-Depth V2 dataset, and tested on 654 images, following the procedure established in [2]. For testing, the depth predictions for are evaluated on a pre-defined center cropping by Eigen [13].

5.3.3 Performance Evaluation Metrics

Each tested network in this chapter is evaluated using several error and accuracy metrics that were used in prior works [2, 13, 15, 29]. More specifically, the following performance metrics were leveraged:

- relative absolute error (Abs Rel): $\frac{1}{N}\sum_{i=1}^{N}\frac{|y_i-\hat{y}_i|}{\hat{y}_i}$
- root mean squared error (rmse): $\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i-\hat{y}_i)^2}$
- average log error (Log10): $\frac{1}{N}\sum_{i=1}^{N}|log(y_i)-log(\hat{y}_i)|$
- δ_i accuracy: % of y_i s.t. $max(\frac{y_i}{\hat{y}_i},\frac{\hat{y}_i}{y_i}) = \delta < thr$ for $thr = 1.25, 1.25^2, 1.25^3$

where y_i is a pixel in predicted depth image, $\hat{y}_i$ is a pixel in the ground-truth depth image and N is the total number of pixels in the depth image.

Finally, evaluating the real-world performance of the proposed DepthNet Nano in a realistic embedded scenario, inference speed and power efficiency evaluations on a Jetson AGX embedded module at two different power budgets, 30W and 15W, are performed. Since OLIV is also to be able to run on a CPU device, the inference speed on a 2.3 GHz Dual-Core Intel Core i5 CPU is evaluated. For inference speed, the framerate (FPS) is computed, while for power efficiency the number of images processes per sec per watt (i.e., images/sec/watt) is computed.

5.4 Architectural Design

The network architecture of the proposed DepthNet Nano, which is illustrated in Fig. 5.1, has several interesting characteristics that are discussed in detail below.

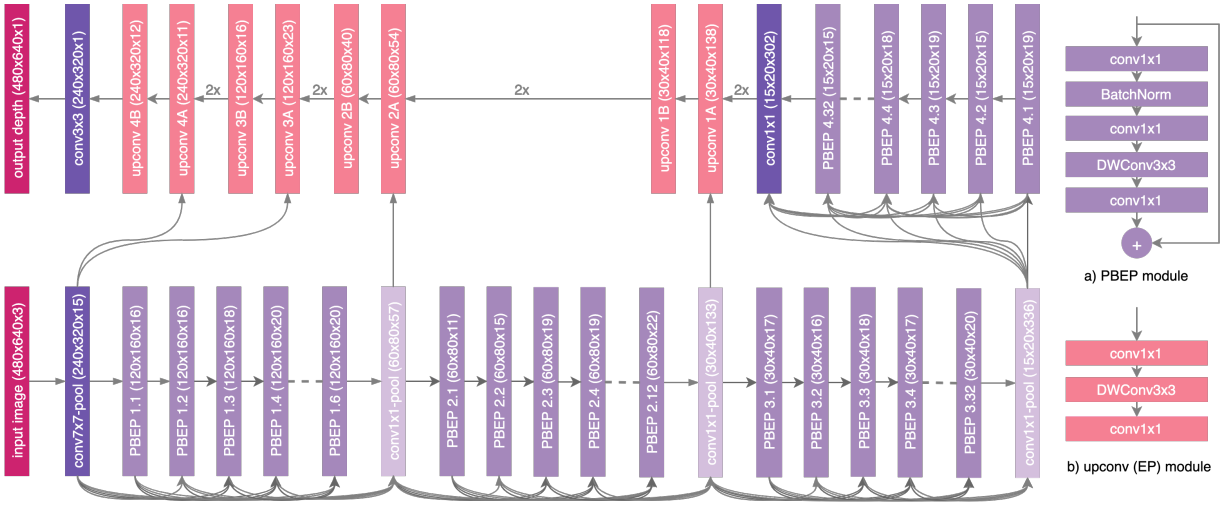


Figure 5.1: DepthNet Nano Architecture. The network architecture exhibits high macroarchitecture heterogeneity with a mix of PBEP and EP modules, as well as individual 7×7, 3×3, and pointwise convolution layers. Furthermore, the architecture exhibits high microarchitecture heterogeneity. Finally, the network architecture possesses a very deep densely-connected self-normalization macroarchitecture, which has not been previously explored.

5.4.1 Self-Normalization Macroarchitecture

The first interesting characteristic of the DepthNet Nano architecture is its self-normalizing property within a very deep densely-connected encoder-decoder network architecture, which has not been previously explored. More specifically, rather than leveraging popular activation functions such as Rectifier Linear Units (ReLU) that are more commonly found in depth estimation networks, the proposed DepthNet Nano architecture heavily leverages Scaled Exponential Linear Units (SELU) [26] as the only form of activation within the network architecture, which can be defined as

$$\text{selu}(x) = \lambda \begin{cases} x & \text{if } x > 0 \\ \alpha \exp(x) - \alpha & \text{if } x \leq 0 \end{cases} \tag{5.1}$$

One of the key advantages of SELUs is their self-normalizing properties that makes learning more robust for deep neural networks. More specifically, since the activations in each layer of the network are close to zero mean and unit variance, as the batched features propagate through, the distribution of the features will converge towards zero mean and unit variance, which makes the values of different features comparable. The self-normalizing property is achieved with SELUs by decreasing the variance for negative inputs and increasing the variance for positive inputs. To achieve zero mean and unit variance, the amount of decrease for very negative inputs and the amount of increase for near zero values are greater than other inputs.

In Table 5.1, the performance of RELU and SELU activations are compared based on accuracy and relative absolute error. For both module types, which will be further explain in the next section, SELU achieves higher performance. The result shows that the use of SELUs in DepthNet Nano is able to achieve higher depth estimation accuracy and lower error than with the use of RELUs, despite high network inter-connectivity.

Table 5.1: DepthNet Nano performance on accuracy, error, computational complexity and size based on RELU or SELU activation and PBConv or PBEP module type.

Activation	Module	$\delta_1 < 1.25$ (higher is better)	Abs Rel (lower is better)	MACs [G]	Params [M]
RELU	PBConv	0.821	0.139	7.39	4.48
	PBEP	0.816	0.140	4.40	3.46
SELU	PBConv	0.825	0.137	7.39	4.48
	PBEP	0.816	0.139	4.40	3.46

5.4.2 Densely Connected Projection BatchNorm Expansion Projection Macroarchitecture

Another interesting characteristic of the DepthNet Nano architecture is the densely connected projection batchnorm expansion projection (PBEP) module, which are leveraged heavily in the encoding layers of the network architecture (see Fig. 5.1). Compared to the expansion projection (EP) modules leveraged extensively in the decoding layers of the network architecture (see Fig. 5.1), which have been seen in other literature on efficient network architectures [44, 48, 8], PBEP has an additional projection layer that decreases

the number of channels of the previous layer before expanding the layer for depth-wise convolution. PBEP macroarchitecture consists of:

1. A projection layer, where the output channels of the previous layer are projected to a lower dimensionality in this layer using 1×1 convolutions.
2. A batch normalization layer that normalizes the output of a previous layer to improve the stability of the network.
3. An expansion layer, where 1×1 convolutions are leveraged to expand the output of the batch normalization layer to a higher dimensionality.
4. A depth-wise convolution layer, where spatial convolutions with a different filter are applied to each of the individual output channels of the expansion layer.
5. A projection layer with 1×1 convolutions that projects the output channels from the depth-wise convolution layer to a lower dimensionality.

In Table 5.1, the PBEP module is compared with the PBConv module, shown in Fig. 5.2, based on accuracy, error computational complexity and size. The PBConv module can be found in the initial network design prototype [2] and consists of a projection layer, followed by a batch normalization layer with a type of activation, and finally a 3×3 convolutional layer. Table 5.1 shows that although the use of PBEP modules result in lower accuracies and higher errors compared to PBConv modules, there are significantly less multiply add accumulations, which corresponds to faster compute times. PBEP modules also reduce the number of parameters. The use of densely connected PBEP macroarchitectures reduces the architectural and computational complexity of the DepthNet Nano architecture while maintaining high model expressiveness and producing high quality depth estimations.

5.4.3 Macroarchitecture and Microarchitecture Heterogeneity

Unlike hand-crafted architectures, the generated macroarchitecture and microarchitecture within the network can differ greatly from layer to layer. There are a mix of different type of modules, such as PBEP and EP modules, as well as individual 7×7, 3×3 and 1×1 convolution layers. In addition, the same module type has vastly different microarchitectures since each module is catered specifically for the needs of the task. For instance, each PBEP and EP module have different numbers of channels to represent the learned features and a different multiplicity for the channel expansion layer.

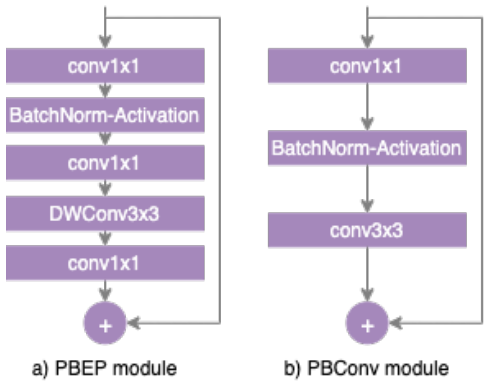


Figure 5.2: PBEP vs PBConv module. a) PBEP module consists of a projection layer, followed by a batch normalization layer with a form of activation (RELU/SELU), an expansion layer, a depth-wise convolution layer and another projection layer. b) PBConv module consists of a projection layer, followed by a batch normalization layer with a form of activation and a 3×3 convolution layer.

The benefit of high macroarchitecture and microarchitecture heterogeneity in DepthNet Nano network architecture is that it enables each component of the network architecture to be uniquely tailored to achieve a very strong balance between architectural and computation complexity while maintaining model expressiveness. The architectural diversity in DepthNet Nano demonstrates the advantage of leveraging a human-collaborative design strategy as it would be difficult for a human designer, or other design exploration methods to customize a network architecture to the same level of architectural granularity.

5.5 Experimental Results

To demonstrate the efficacy of the proposed DepthNet Nano network designed using the human-machine collaborative design strategy, its network architecture complexity, depth estimation performance, and computational cost on the NYU-Depth V2 dataset are examined. In addition to quantitative analysis, a qualitative analysis on the predicted depth maps is also conducted.

5.5.1 Quantitative Analysis

To study the efficacy of human-machine collaborative design, DepthNet Nano is evaluated alongside state-of-the-art depth estimation networks on performance metrics defined in Section 5.3.3, network architecture complexity, and computational complexity on the NYU Depth V2 dataset, as shown in Table 5.2, respectively. Since this study targets high quality depth estimation, only state-of-the-art networks in literature with δ_1 accuracies greater than 0.8 on NYU Depth v2 are compared.

DepthNet Nano has significantly lower architecture complexity and computational complexity compared to the tested state-of-the-art networks. For example, DepthNet Nano has 18 times fewer parameters and requires 9.7 times fewer MAC operations for inference than [29], while achieving comparable δ_1, δ_2 and δ_3 accuracies. Furthermore, it is important to note that DepthNet Nano has achieved these significant computational complexity improvements despite the fact that the input image size is 2 times larger, both vertically and horizontally, compared to [29].

Table 5.2: Performance on NYU Depth V2. All networks are evaluated using a pre-defined center cropping [13]. Best results in **bold**.

Model	Input Size	MACs [G]	Params [M]	higher is better			lower is better		
				$\delta_1 < 1.25$	$\delta_2 < 1.25^2$	$\delta_3 < 1.25^3$	Abs Rel	RMSE	Log10
Laina et al. [29]	228×304	42.7	63.6	0.811	0.953	0.988	0.127	0.573	0.055
Fu et al. [15]	257×353	60.2	110.3	0.828	0.965	0.992	**0.115**	0.509	**0.051**
Alhashim et al. [2]	480×640	49.2	21.52	**0.846**	**0.974**	**0.994**	0.123	**0.465**	0.053
DepthNet Nano	480×640	**4.4**	**3.46**	0.816	0.958	0.989	0.139	0.599	0.059

Table 5.3: NYU Depth V2 Inference. All networks are evaluated on a Jetson AGX Xavier embedded module and 2.3 GHz Dual-Core Intel Core i5 CPU. Best results in **bold**.

Model	MACs [G]	30W [FPS]	15W [FPS]	30W [$\frac{\text{images/s}}{\text{watt}}$]	15W [$\frac{\text{images/s}}{\text{watt}}$]	CPU [FPS]
Alhashim et al. [2] (480×640)	49.2	6.83	3.76	0.228	0.251	0.74
DepthNet Nano (480×640)	**4.4**	**15.8**	**8.8**	**0.527**	**0.587**	**1.8**
Alhashim et al. [2] (224×224)	8.04	–	–	–	–	2.96
DepthNet Nano (224×224)	**0.72**	–	–	–	–	**9.35**

The inference speed of DepthNet Nano and [2] at the initial resolution of 480×640 are first compared on a Jetson AGX Xavier embedded module in Table 5.3. DepthNet Nano is more than 2.3 times faster and more energy efficient than Alhashim et al. [2] at both 30W and 15W power budgets. In addition, inference results at the input resolution of 224×224 for OLIV on a 2.3 GHz Dual-Core Intel Core i5 CPU further indicate the

efficiency of DepthNet Nano. At the lower resolution of 224×224, DepthNet Nano is more than 3 times faster compared to [2] at the same resolution. These quantitative results demonstrate that the proposed DepthNet Nano networks, created using a human-machine collaborative design strategy, can achieve a strong balance between accuracy, network architecture complexity, and computational complexity that makes it very well suited for embedded depth estimation for edge scenarios.

5.5.2 Qualitative Analysis

In additional to quantitatively evaluating the performance DepthNet Nano, a qualitative analysis is also conducted to present areas that may be not be evaluated by the error metrics or inference speeds. Figure 5.3 shows three examples from the NYU-Depth V2 dataset. Each RGB image has a corresponding ground-truth dense depth map and predicted dense depth maps from [2] and DepthNet Nano. A number of interesting observations can be made based on the produced dense depth maps.

First of all, the dense depth maps produced by DepthNet Nano are only slightly less detailed than that produced by [2] for NYU Depth V2, despite DepthNet Nano requiring approximately 10 times fewer MAC operations. Another interesting observation is the first row in Figure 5.3. In the ground-truth depth map, the shower curtain reflected in the mirror has a different depth than the mirror. Similarly, [2] also predicted a different depth for the reflected shower curtain. However, the proposed DepthNet Nano predicted a more consistent depth within the mirror. In this case, although the predicted depth maps for the mirror visually appears to be more accurate for DepthNet Nano, the result is not reflected when calculating error since the ground-truth depth map has an inconsistent depth for the mirror. In Fig. 5.4, the results of DepthNet Nano for the office scene with cluttered objects that are further away are analyzed. Based on the qualitative analysis, [2] does a better job with fine-grain edges. Since DepthNet Nano has significantly fewer parameters, the edges of the cluttered objects are not well distinguished. Overall, visually, DepthNet Nano produced dense depth maps comparable to state of the art.

5.6 Summary

In this chapter, DepthNet Nano, a highly compact self normalizing network that is tailored and designed using a human machine collaborative design strategy is proposed for OLIV.

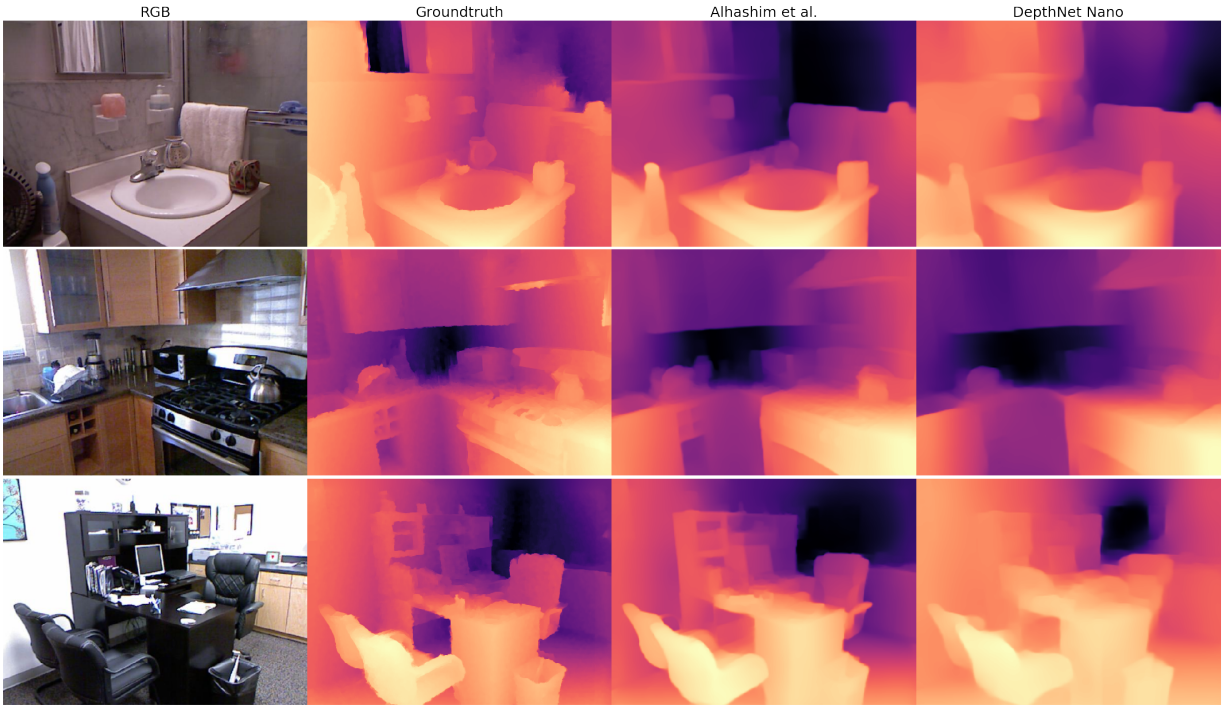


Figure 5.3: Visualized results on NYU Depth V2 dataset [46]. (Left to right) input RGB image, ground truth, and depth estimations from Alhashim et al. [2] and DepthNet Nano. DepthNet Nano was able to produce high-quality depth estimations despite requiring 10× fewer MAC operations than Alhashim et al. [2].

By coupling human-driven principled network design prototyping and machine-driven design exploration, the resulting DepthNet Nano network architecture exhibited highly customized macroarchitecture and microarchitecture designs, as well as self-normalizing characteristics that provide a strong balance between architecture complexity, computational complexity, and depth estimation performance. Experimental results across the NYU-Depth V2 dataset demonstrated that the proposed DepthNet Nano possesses a significantly more architecturally and computationally efficient network architecture compared with state-of-the-art networks while achieving comparable performance. Furthermore, experiments demonstrate that DepthNet Nano has significantly faster inference speeds and energy efficiency on the Jetson AGX Xavier embedded module. DepthNet Nano is also able to achieve 9.3 FPS on a 2.3 GHz Dual-Core Intel Core i5 CPU, which is more than 3 times faster than current state of the art indoor depth estimation models.

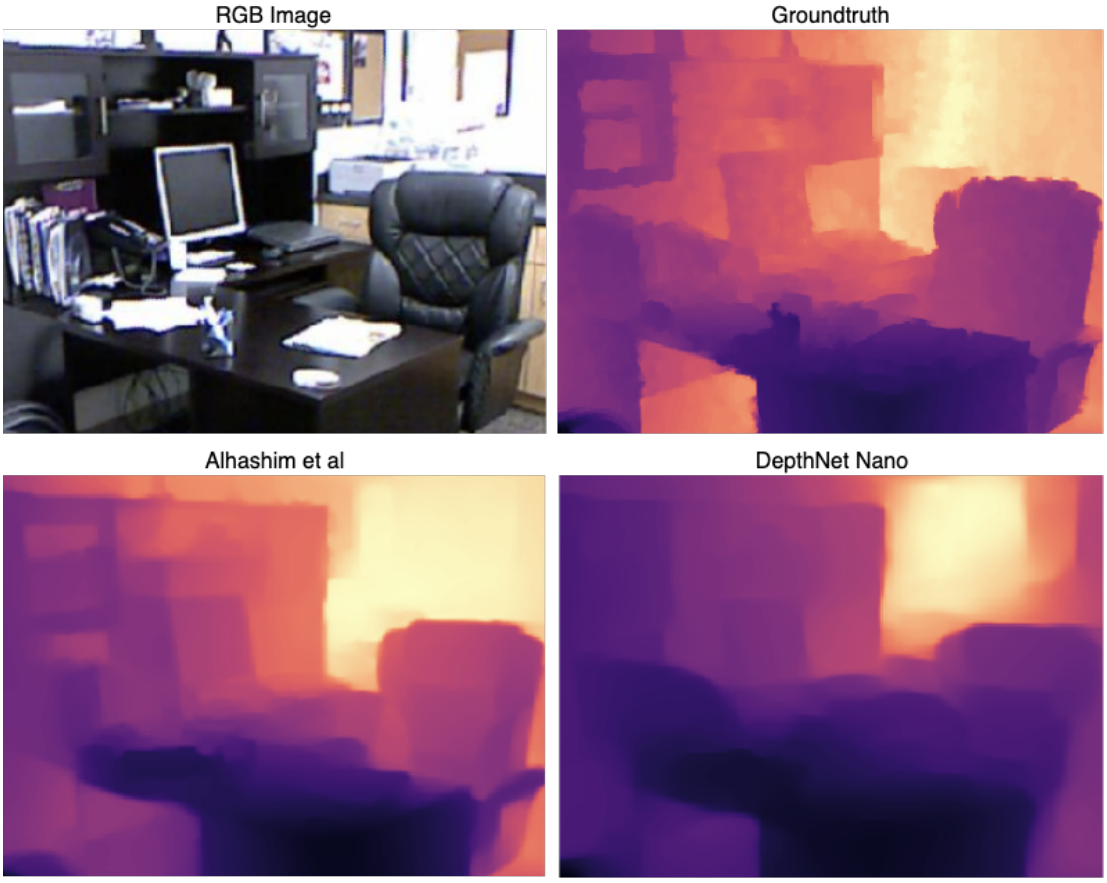


Figure 5.4: Zoomed in view of cluttered office desk from NYU Depth V2 dataset [46]. Comparison of ground truth depth map, and predicted depth map by [2] and DepthNet Nano.

Chapter 6

Conclusion

In this chapter, a brief summary of the thesis and the key contributions are described in Section 6.1. Recommendations of the methods are discussed in Section 6.2, and potential future work for this research is outlined in Section 6.3.

6.1 Summary of Thesis and Contributions

In this thesis, the design of OLIV, a novel human-centered end-to-end artificial intelligence-powered assistant system was proposed to assist individuals with impaired vision in their day to day tasks in locating displaced object in indoor environments. To ensure privacy, OLIV runs on local devices. However, to be able to detect and locate objects in real-time, the object detection and monocular depth estimation networks for OLIV must be efficient. Current deep neural networks are large and require powerful processing units, such as GPUs, which are expensive and not widely adopted among common households. To design efficient object detection and monocular depth estimation neural networks for OLIV, two different architecture design exploration approaches are taken. The efficiency of the designed object detection and monocular depth estimation networks were then tested on two indoor datasets, MSCOCO-OLIV and NYU Depth V2, to simulate the indoor environments where OLIV may be used.

In Chapter 4, a micro-architecture exploration approach to create an object detection network that can run on-device specifically for OLIV is investigated. Based on quantitative architecture exploration analysis in Figure 4.5, the model with the optimal trade-off between accuracy, speed and size, as indicated by the NetScore, for the OLIV system

requirements is MobileNetV2-SSD-1.3-224, which has a width multiplier of 1.3 and input resolution of 224. This result is supported by the experimental results in Figure 4.2. As the width and resolution multiplier increase, the mAP and CPU running time also increase, however, there is a certain point, specifically when the width multiplier passes 1.15 and the input resolution reaches 220, where there are diminishing returns.

In Chapter 5, a human machine collaborative design strategy is taken to tailor a highly compact self normalizing monocular depth estimation network, DepthNet Nano, specifically for OLIV. By coupling human-driven principled network design prototyping and machine-driven design exploration, the resulting network architecture exhibited highly customized macroarchitecture and microarchitecture designs, as shown in Figure 5.1, as well as self-normalizing characteristics that provide a strong balance between architecture complexity, computational complexity, and depth estimation performance. Experimental results in Table 5.2 demonstrated that DepthNet Nano possesses significantly less computational requirements while achieving comparable results on the NYU-Depth V2 dataset with state-of-the-art networks. Furthermore, experiments in Table 5.3 demonstrated that DepthNet Nano has significantly faster inference speeds and energy efficiency both on embedded devices, such as the Jetson AGX Xavier, and on CPU, such as 2.3 GHz Dual-Core Intel Core i5. Qualitatively, the dense depth maps produced by the proposed network in Figure 5.3 are only slightly less detailed than the original network despite requiring fewer MAC operations.

6.2 Recommendations

In this thesis, two different neural network design exploration methods were implemented to design efficient neural networks. Although there is a decrease in accuracy, these efficient neural networks are able to be deployed on device to ensure that users' personal data are not exploited. These neural networks also run in real time on local devices, which can provide users with real time updates. Depending on the application, neural network design exploration methods should be implemented to provide a trade-off between not only accuracy, speed and size, but also the amount of data privacy the user can have.

Although having multiple neural networks require more computational and operational costs, these networks can complement each other. Based on the experimental results, both the object detection and depth estimation networks are limited to the type and size of the object. For instance, MobileNet-SSD-1.3-224 does well in detecting objects such as a monitor, laptop and clock, however are worse at detecting objects such as spoon, knife and handbag. This is due to the fact that there are class imbalances in the dataset, leading

to bias towards some classes than others. Due to the reduction in number of parameters, DepthNet Nano has a difficult time in distinguishing objects that are further away and small. Although the object detection network is limited by the type of object and the depth estimation network is limited by the size of the object, synthesis can allow these two different perception tasks to complement each other.

6.3 Future Work

6.3.1 Real World Testing of OLIV

Although testing the designed object detection network and monocular depth estimation network for OLIV on indoor datasets simulate indoor environments, these networks also need to be tested in real world environments to ensure the robustness of the networks on more unseen data. To ensure that OLIV is useful for the visually impaired, OLIV should also be tested on different levels of visual acuity. [51, 47] showed that different levels of functioning have different perceptions of a solution, as well as a diverse range of benefits and problems.

6.3.2 Improve Detection of Small Objects and Object Bias

Both the object detection network and monocular depth estimation network have a lower accuracy when detecting smaller objects and occluded objects. For instance, in Fig. 3.2, small object , such as the computer mouse, cellphone and bottles, were not able to be detected. In the left image Fig. 4.4, the person and bowl were able to be detected, however, the occluded spoon was not detected. In Fig. 5.4, there are many occluded objects on the desk, however, the estimated depth map was not able to distinguish the edges.

In addition, in Fig. 4.4, the cup is detected in the middle image, however, larger objects such as the mouse and book are not detected. In the right image of Fig. 4.4, the laptop and mouse are detected, however, the handbag, which is larger than the laptop and mouse, is not detected. This suggests that the dataset is imbalanced.

For future work, the goal is to incorporate methods to improve the detection of small objects, as well as to address the inherent class imbalance in the MSCOCO-OLIV dataset. To improve the detection of small object, methods, such as atrous convolutions [5], can

be experimented. To address the class imbalance problem, there are methods, such as augmentations and class balance loss weightings [10], for classes with less samples.

6.3.3 Outdoor Scenes

Currently, both the object detection and depth estimation networks are only trained on objects in indoor environments. However, since the OLIV system can run on mobile devices, OLIV can be extended to outdoor scenes to help assist in a wide range of tasks. In order to be extended to outdoor environments, future work will include training these neural networks on outdoor environments and common objects that can be outdoors in addition to the current MSCOCO-OLIV and NYU-Depth V2 datasets.

6.3.4 Temporal Information

In this thesis, supervised training was used to train the monocular depth estimation network. In recent years, self-supervised training, where there is no labeled data, for monocular depth estimation has shown promise [16, 57]. For future work, strategies for incorporating temporal information into the DepthNet Nano architecture in a way that improves performance while maintaining low architecture and computational complexity will be explored. By incorporating self-supervised techniques along with supervised training, the features learned from self-supervised training can be used to improve the current training method.

6.3.5 Federated Learning

As OLIV gets deployed on local devices, the object detection and depth estimation model can be continuously trained and improved based on the user's environment. In addition to tailoring the neural network for the user's environment, the data the user collects can also be used to train a centralized model. Current machine learning approaches require all training data in one centralized location to train or further improve the neural network. Recently, an approach called *federated learning* enables individual local devices to collaboratively learn a shared prediction model while keeping all the training data on the device [35]. For future work, federated learning will be experimented since this approach allows the centralized model to be continuously improved while the user's data remains on their local device.

References

[1] Tousif Ahmed, Roberto Hoyle, Kay Connelly, David Crandall, and Apu Kapadia. Privacy concerns and behaviors of people with visual impairments. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3523–3532, 2015.

[2] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv preprint arXiv:1812.11941*, 2018.

[3] Margret Baltes, Ineke Maas, Hans-Ulrich Wilms, Markus Borchelt, and Todd Little. *Everyday Competence in Old and Very Old Age: Theoretical Considerations and Empirical Findings*, pages 384–402. 01 1999.

[4] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson, editors, *Proceedings of the 1st Conference on FAT*, volume 81 of *PMLR*, pages 77–91, 23–24 Feb 2018.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[8] Xiangxiang Chu, Bo Zhang, Ruijun Xu, and Jixiang Li. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. *arXiv preprint arXiv:1907.01845*, 2019.

[9] Patrick Clary. Lookout: an app to help blind and visually impaired people learn about their surroundings, May 2018.

[10] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019.

[11] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[13] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[14] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[15] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[16] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3838, 2019.

[17] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1135–1143. Curran Associates, Inc., 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[20] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.

[21] Jonathan Huang, Vivek Rathod, Ronny Votel, Derek Chow, Chen Sun, Menglong Zhu, Alireza Fathi, and Zhichao Lu. Tensorflow object detection api. https://github.com/tensorflow/models/tree/master/research/object_detection, 2018.

[22] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[23] Steven Kelley. Seeing ai: Artificial intelligence for blind and visually impaired users, 2019.

[24] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2017.

[25] Florian Kirchbuchner, Tobias Grosse-Puppendahl, Matthias R. Hastall, Martin Distler, and Arjan Kuijper. Ambient intelligence from senior citizens' perspectives. In *Ambient Intelligence*, pages 48–59, Cham, 2015. Springer International Publishing.

[26] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in neural information processing systems*, pages 971–980, 2017.

[27] B. F. Klare, M. J. Burge, J. C. Klontz, R. W. Vorder Bruegge, and A. K. Jain. Face recognition performance: Role of demographic information. *IEEE Transactions on Information Forensics and Security*, 7(6):1789–1801, Dec 2012.

[28] Juliane Köberlein, Karolina Beifus, Corinna Schaffert, and Robert P Finger. The economic burden of visual impairment and blindness: a systematic review. *BMJ Open*, 3(11), 2013.

[29] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016.

[30] Chaiwoo Lee and Joseph F. Coughlin. Perspective: Older adults' adoption of technology: An integrated approach to identifying determinants and barriers. *Journal of Product Innovation Management*, 32(5):747–759, 2015.

[31] M. Leo, G. Medioni, M. Trivedi, T. Kanade, and G.M. Farinella. Computer vision for assistive technologies. *Computer Vision and Image Understanding*, 154:1 – 15, 2017.

[32] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[33] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015.

[34] Kirsten Lloyd. Bias amplification in artificial intelligence systems. *CoRR*, abs/1809.07842, 2018.

[35] Brendan McMahan and Daniel Ramage. Federated learning: Collaborative machine learning without centralized training data. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html, 2017.

[36] George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998.

[37] World Health Organization. Blindness and vision impairment. https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment, 2019.

[38] Siyuan Qiao, Zhe Lin, Jianming Zhang, and Alan L Yuille. Neural rejuvenation: Improving deep network training by enhancing computational resource utilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 61–71, 2019.

[39] Inioluwa Deborah Raji and Joy Buolamwini. Actionable auditing: Investigating the impact of publicly naming biased performance results of commercial ai products. In *Conference on AIES*, 2019.

[40] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[42] Mark O. Riedl. Human-centered artificial intelligence and machine learning. *CoRR*, abs/1901.11184, 2019.

[43] D. A. Ross. Implementing assistive technology on wearable computers. *IEEE Intelligent Systems*, 16(3):47–53, May 2001.

[44] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[45] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2. https://github.com/tensorflow/models/blob/master/research/slim/nets/mobilenet, 2018.

[46] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012.

[47] Sylvia Söderström and Borgunn Ytterhus. The use and non-use of assistive technologies from the world of information and communication technology by visually impaired young people: a walk on the tightrope of peer inclusion. *Disability & Society*, 25(3):303–315, 2010.

[48] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. *arXiv preprint arXiv:1807.11626*, 2018.

[49] Tensorflow. Profile model. https://www.tensorflow.org/api_docs/python/tf/compat/v1/profiler/profile, 2020.

[50] Yingli Tian, Xiaodong Yang, Chucai Yi, and Aries Arditi. Toward a computer vision-based wayfinding aid for blind persons to access unfamiliar indoor environments. *Machine vision and applications*, 24:521–535, 04 2013.

[51] D. Townsend, F. Knoefel, and R. Goubran. Privacy versus autonomy: A tradeoff model for smart home monitoring technologies. In *2011 IEEE EMBC*, pages 4749–4752, Aug 2011.

[52] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. Fastdepth: Fast monocular depth estimation on embedded systems. *arXiv preprint arXiv:1903.03273*, 2019.

[53] Alexander Wong. Netscore: Towards universal metrics for large-scale performance analysis of deep neural networks for practical usage. *CoRR*, abs/1806.05512, 2018.

[54] Alexander Wong, Mahmoud Famuori, Mohammad Javad Shafiee, Francis Li, Brendan Chwyl, and Jonathan Chung. Yolo nano: a highly compact you only look once convolutional neural network for object detection. *arXiv preprint arXiv:1910.01271*, 2019.

[55] Alexander Wong, Mohammad Javad Shafiee, Brendan Chwyl, and Francis Li. Ferminets: Learning generative machines to generate efficient neural networks via generative synthesis. *arXiv preprint arXiv:1809.05989*, 2018.

[56] Tien-Ju Yang, Andrew Howard, Bo Chen, Xiao Zhang, Alec Go, Mark Sandler, Vivienne Sze, and Hartwig Adam. Netadapt: Platform-aware neural network adaptation for mobile applications. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 285–300, 2018.

[57] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2018.